T77-10766 NMR
E80-10218
JSC-12537
NASA CR-
160641

AS BUILT DESIGN SPECIFICATION FOR THE YIELD ESTIMATION SUBSYSTEM (YES) MONTHLY YIELD DATA BASE AND SUPPORTING PROGRAMS

Job Order 74-963

AD 63-1347-4963-01

(E80-10218) AS-BUILT DESIGN SPECIFICATION FOR THE YIELD ESTIMATION SUBSYSTEM (YES) MONTHLY YIELD DATA BASE AND SUPPORTING PROGRAMS (Lockheed Electronics Co.) 124 p HC A06/MF A01 CSCL 05B G3/43

N80-29795

Unclas 00218

Prepared By

Lockheed Electronics Company, Inc.
Systems and Services Division
Houston, Texas

Contract NAS 9-15200

For

EARTH OBSERVATIONS DIVISION

National Aeronautics and Space Administration

LYNDON B. JOHNSON SPACE CENTER

Houston, Texas

February 1977

LEC-10034

AS BUILT DESIGN SPECIFICATION FOR THE YIELD ESTIMATION SUBSYSTEM (YES) MONTHLY YIELD DATA BASE AND SUPPORTING PROGRAMS

Job Order 74-963

AD 63-1347-4963-01

PREPARED BY

D. Cook
C. Slemons

APPROVED BY

P. L. Krumm, Supervisor
Software Development Section

Prepared By

Lockheed Electronics Company, Inc.

For

Earth Observations Division

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
HOUSTON, TEXAS

February 1977

CONTENTS

1. SCOPE

This document describes the monthly weather and yield data base and associated computer programs installed on the 360/195 complex at Suitland, Maryland. The system is in support of Yield Estimation efforts of LACIE.

2. APPLICABLE DOCUMENTS

AD 63-1347-4963-01

AD 63-1347-4963-04

AD-04 requires specification for the India data base. This is not available at the time of preparation of this document. Documentation for India will be delivered separately.

3. SYSTEM DESCRIPTION

The monthly yield data base system consists of three components. The first is the computer hardware necessary to support the system. This is described in section 3.1. The second is a data base structure. This is described in section 3.2. The third is a set of support programs. This is described in section 3.3.

3.1 HARDWARE DESCRIPTION

These programs and data are resident on the IBM 360/195 complex at Suitland, Maryland. They should be transferable to any IBM 360-370 series machine with sufficient disk to handle the data base and main memory to support the PL/I optimizing compiler.

3.2 DATA BASE STRUCTURE

The data base (Monthly Yield Data Base) is a tree structure, nodes being countries, regions, districts, etc. Nodes are referred to as levels in the remainder of this document. The basic unit of information is a block. Blocks are of four types: Control, Directory, Data Descriptor and Data, and Model Definition, each with a corresponding PL/I structure given in appendix A.

3.2.1 DATA BASE STORAGE REQUIREMENTS

The data base currently occupies 288 6440 byte blocks, partitioned into three data sets: USA 114 blocks, USSR/Canada 114 blocks, Argentina/Australia 60 blocks.

3.2.2 CONTROL BLOCKS

There is only one control block on a file. It is the first block to be defined and contains information on the block type of every other block in the file. It also contains the location of the directory entry for every level-one region (usually a country).

Control block information is divided into eleven sections, some of which are arrays with subsections.

1. The first section is the file identification name which is a name up to eight characters in length describing the file.
2. The second section gives the number of passwords which are available to use the programs accessing the file.
3. The third section is an array of one to eight passwords, each up to eight characters in length. Any one of the passwords can be used to access the programs. The number of passwords in this section should equal the number given in section 2.
4. The fourth section gives the number of levels in which the data is arranged.
5. The fifth section is an array of one to eight level names, each up to 24 characters in length. The levels refer to the organization of the data. The smaller the level number, the larger the region; the larger the level number, the smaller the region. For example, level one is probably a country, whereas level four may be a crop reporting district. Data are collected at the smaller regions (higher level numbers) and may or may not be aggregated up to lower level numbers. The number of level names in this section should equal the number given in section 4.
6. The sixth section gives the number of codes, not to exceed 32, for variables which are in the data blocks.

7. The seventh section is an array with subsections giving information on each of the codes. The six subsections are repeated for each of the codes, the number of which should equal the number in section 6.

   a. The code number identifies the variable, for example, precipitation.

   b. The unit number identifies how the variable is measured, for example, millimeters.

   c. The base is the number of digits allowed for an observation.

   d. The scale is the power of ten by which the observation is multiplied. This may be simply the number of decimal places in the observation; it eliminates keypunching the decimal points.

   e. The code name is a name up to 24 characters in length associated with the code number.

   f. The unit name is a name up to 24 characters in length associated with the unit number.

8. The eighth section gives the number of level-one regions on the file. This will probably be the number of countries, and cannot exceed 24.

9. The ninth section is an array with subsections giving information on each of the level-one regions. The five subsections are repeated for each level-one region, the number of which should equal the number in section 8.

   a. The code number identifies the level-one region.

   b. The number of directories is the current count of directory entries on the file for that level-one region and all higher level regions within that level-one region.

c. The record number is the location on the file of the directory block containing the directory entry for the level-one region.

d. The displacement number is the position in the directory block where the directory entry for the level-one region begins.

e. The level-one region name is the name up to 24 characters in length for that region.

10. The tenth section gives the number, not exceeding 601, of records or blocks which the file can contain, excluding the control block. Each block is 6440 bytes long.

11. The eleventh section is an array with subsections giving information on each of the records on the file. The three subsections are repeated for each record, the number of which should be equal to the number in section 10.

a. The record type identifies each record according to what kind of block it contains.

1) A type of 0 (zero) means blank or no information recorded on the record.

2) A type of -1 (negative one) means the record contains directory entries.

3) A type of +1 (positive one) means the record contains a data descriptor and data.

4) A type of +2 (positive two) means the record contains a model definition block.

b. The free space is the number of bytes on the record which are blank.

c. The location is the position on the record where the free space begins.

3.2.3 DIRECTORY BLOCKS

There is a directory entry for every level, sublevel, sub-sublevel, etc., to a maximum of eight levels. The entries contain information which gives the location of other entries at the same level and at the next higher and next lower levels, and also information which gives the location in the file of the entry's data descriptor and model definition. Directory entries are grouped together in directory blocks, with the number of blocks dependent on the number of reporting districts.

A directory entry is divided into fifteen sections, one of which is an array with three subsections. A directory block contains up to 84 directory entries, each 76 bytes long, for a level-one region. More than one directory block may be needed for a level-one region, but a directory block does not contain directory entries for more than one level-one region.

1. The first section is the level number for the entry. It ranges from one to the maximum number of levels defined in the fourth section of the control block.

2. The second section is the code number for the entry. It is a unique number only within that particular sublevel. For example, there could be a code number of 10 for more than one level-three entry provided each of them is associated with a different level-two region.

3. The third section is the latitude for the region. It is a positive number for regions in the northern hemisphere and negative for those in the southern hemisphere. For large areas it is the latitude of some central point.

4. The fourth section is the longitude for the region. It is a positive number in the western hemisphere and negative in the eastern hemisphere. For large areas it is the longitude of some central point.

5. The fifth section is the name of the region to which the directory entry pertains.

6. The sixth section is the location on the file of the directory block which contains the directory entry for the "parent" of the current entry. The "parent" of any entry is the entry with the next smallest level number and of which the original entry is a part. For example, the Black Lands is a level-four region whose parent is the level-three region, Texas. The parent of Texas is the level-two region, the Great Plains, whose parent is the level-one region, the United States. Level-one regions have no parent, so the location is coded as -1 (negative one).

7. The seventh section is the position within the directory block where the parent's directory entry begins. The directory block location is given in section 6; if the directory block location is a -1 (negative one), this position is set to a +1 (positive one).

8. The eighth section is the location on the file of the directory block which contains the directory entry for the "brother" of the current entry. The "brother" of any entry is the entry with the same level number, the same parent, and the next largest code number. For example, the brother of the Black Lands with code number 40 is East Texas North with code number 51. Both are at level four and have Texas as their parent. The brother of East Texas North is East Texas South which has the code number 52. The last entry under a given parent has no brother, so the location is coded as a -1 (negative one).

9. The ninth section is the position within the directory block where the brother's directory entry begins. The directory block location is given in section 8; if the directory block location is -1 (negative one), this position is set to a +1 (positive one).

10. The tenth section is the location on the file of the directory block which contains the directory entry for the "child" of the current entry. The child of any entry is the entry with the next largest level number and the smallest code number of all entries which are a part of the current entry. For example, the North High Plains, which is at level four and has a code number of 11, is the child of Texas. The entries with the highest level numbers have no children, so the location is coded as a -1 (negative one).

11. The eleventh section is the position within the directory block where the child's directory entry begins. The directory block location is given in section 10; if the directory block location is a -1 (negative one), then this position is set to a +1 (positive one).

12. The twelfth section is the location on the file of the block which contains the data descriptor entry, followed immediately by the data associated with the directory entry. If there are no data for the entry, this location is coded as a -1 (negative one).

13. The thirteenth section is the position within the data descriptor and data block where the data descriptor entry begins. The data block location is given in section 12; if the data block location is a -1 (negative one), then this position is coded as a +1 (positive one).

14. The fourteenth section is a ten-digit code number which is unique for every directory entry. It is made up of the code numbers for all lower level regions of which the particular region is a part, and the region's own code number. The first two digits contain the level-one region code, the second two contain the level-two region code, etc. When the region's own code is reached, the remaining digits are coded as zeros. For example, the United States would be

coded as 0300000000, the Great Plains as 0301000000, Texas as 0301480000, and the Black Lands as 0301484000.

15. The fifteenth section is an array with subsections giving information on the model definition blocks for up to four different crops.

    a. The crop code identifies the crop whose yield the model is estimating.

    b. The model record number gives the location on the file of the model definition block for the particular crop and region.

    c. The model displacement number gives the position in the block where the model definition begins.

3.2.4 DATA DESCRIPTOR AND DATA BLOCKS

There is a data descriptor entry preceding the data for every region for which data is available. It contains information about the region and completely describes the amount, type, and format of the data that follows. The data include historic weather and yield measurements for a particular region.

A data descriptor entry, which is 336 bytes long, is divided into fifteen sections, one of which is an array with five subsections. The data descriptor entry immediately precedes the data for all years from a certain region. In many cases, there will be only one region's descriptor and data on a 6440-byte record. However, if there are a limited number of variables recorded and/or a limited number of years available, a second region's descriptor and data may be started in the middle of the record at byte 3221.

1. The first section is the identification number. It is the same ten-digit code number which is given in section fourteen of the region's directory entry and has been previously described in part 3.2.3.

2. The second section is the World Meteorological Organization's code number for the region. If the region has no WMO number, this section is coded as zero.

3. The third section is the latitude of the region and is identical to the third section of the region's directory entry.

4. The fourth section is the longitude of the region and is identical to the fourth section of the region's directory entry.

5. The fifth section is the elevation of the region. If unknown, this is coded as zero.

6. The sixth section is the total number of years of data from the particular region which a record (or half a record) could contain. It will depend on the amount of data recorded for each year, which will vary according to country.

7. The seventh section is the current count of the number of years of data from the particular region that the record contains.

8. The eighth section is the length in bytes needed to store one year's data. This should be the same for regions within a country, but will vary between countries.

9. The ninth section is the location on the file of the data block that contains the first chronological year's data for the region. In most cases this should be the same record location as the data descriptor entry's location. Also in most cases, the first chronological year and the first physical year in the data block are the same.

10. The tenth section is the position in the data block where the first chronological year's data begin.

11. The eleventh section is the location on the file of the data block that contains the last chronological year's data for the region. In most cases the last chronological year and the last physical year in the data block are the same.

12. The twelfth section is the position in the data block where the last chronological year's data begin.

13. The thirteenth section is reserved space, eighteen bytes long. It is coded as blank and can be used later if needed.

14. The fourteenth section is the number of codes, not to exceed twelve, for variables used in the data which follow.

15. The fifteenth section is an array with subsections giving information on each of the codes. The five subsections are repeated for each of the codes, the number of which should equal the number in section 14. The codes in the data descriptor entry should be a subset of the codes in the control block.

    a. The code number identifies the variable. The code number table is given in appendix B.

    b. The number of elements is the number of times the variable is recorded in a year. For example, if precipitation is recorded on a monthly basis, the number of elements is twelve.

    c. The element size is the length in bytes of a single observation of the variable. For example, the precipitation for a given month uses two bytes of storage.

    d. The number of subcodes is the number of subdivisions into which the variable is broken down. For example, the variable production can be broken down into production for spring wheat and for winter wheat.

    e. An array of one to eight code numbers identifies the subdivisions of the variable. The number of codes in this array should equal the number given in part d above.

The data which are stored in the data descriptor and data block will vary from country to country. However, for all countries the data for a region are grouped according to year and begin immediately after the region's data descriptor entry. Also for all countries, the first eight bytes of each year's data will contain the same variables.

1. The first variable is the year in which the data were recorded.
2. The second variable is the location on the file of the data block that contains the next chronological year's data for the region.
3. The third variable is the position in the data block where the next chronological year's data begin.
4. The fourth variable is two bytes of reserved space which is coded as blank and can be used later if needed.

3.2.5 MODEL DEFINITION BLOCKS

There is a separate model definition block for every district requiring a unique yield model. It contains the information needed to run the appropriate model for that district.

3.3 SUPPORTING PROGRAMS

There are three classes of programs supporting the data base:

1. Initialization and Definition Programs
   (INITIAL and YESM001, 3.3.1 to 3.3.10) These prepare the data base and subsections of the data base for data entry.
2. Data Entry Programs
   (Loaders and UPDDATA, 3.3.11 to 3.3.12) These programs load data into the data base.

3. Listing Programs
(LISTJOB, YESLS02, YESLS04, 3.3.13 to 3.3.15) These programs list data stored in the data base.

3.3.1 DATA BASE INITIALIZATION PROGRAM (INITIAL)

INITIAL prepares the data base for entry of directory and data by setting size information parameters and filling the data area with zeros.

3.3.1.1 Linkages

None.

3.3.1.2 Interfaces

INITIAL must be run first.

3.3.1.3 Inputs

The file name via JCL. The file size encoded at line 430. (See listing.)

3.3.1.4 Outputs

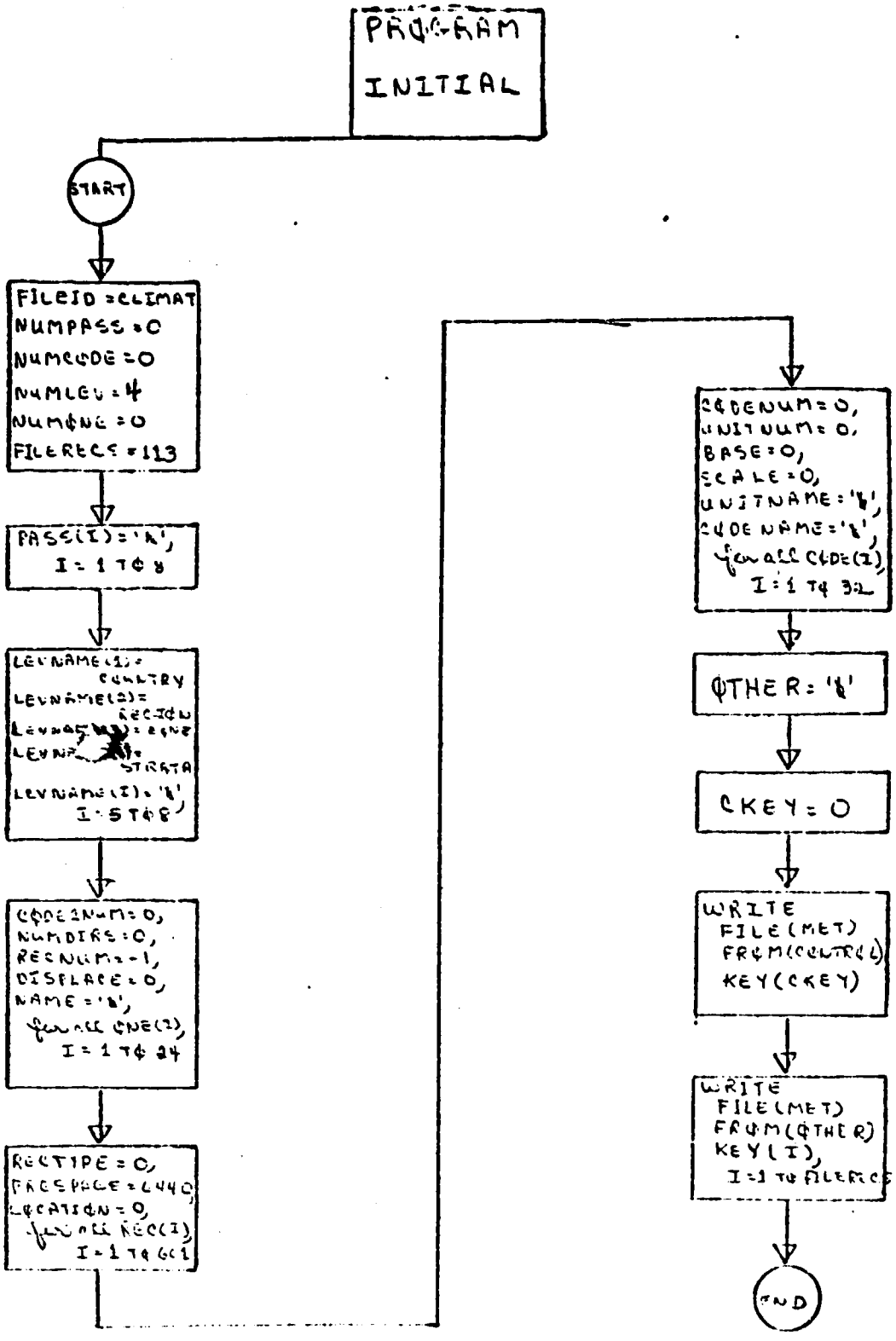
Data base prepared for subsequent processing.

3.3.1.5 Flow Chart

Next page.

3.3.1.6 Listing

Follows flow chart.



ORIGINAL PAGE IS
OF POOR QUALITY

```
RUN NO. 15      DATE 11/12/75    TIME 0910          LISTING OF MODULE INITAD
DESCRIPTION     DATA BASE PGM

MASTER FILE          W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER      10/13/75
LAST DATE COPIED       NONE
LAST UPDATE            NONE

PASSWORD             JVSV
PROGRAMMER           LEC
LANGUAGE             PLI
PROC PARAMETER       $PROJCL

INITAL: PROC OPTIONS(MAIN);
        DCL 1 CONTROL,
              2 FILEID CHAR(7),
              2 NUMPASS FIXED BIN(15,0),
              2 PASS(2) CHAR(4),
              2 NUMLEV FIXED BIN(15,0),
              2 LEVNAME(4) CHAR(24),
              2 NUMCODE FIXED BIN(15,0),
              2 CODE(12),
                3 CODENUM FIXED BIN(15,0),
                3 UNITNUM FIXED BIN(15,0),
                3 BASE FIXED BIN(15,0),
                3 SCALE FIXED BIN(15,0),
                3 CODENAME CHAR(24),
                3 UNITNAME CHAR(24),
              2 NUMONE FIXED BIN(15,0),
              2 ONE(24),
                3 CODENUM FIXED BIN(15,0),
                3 NUMDISS FIXED BIN(15,0),
                3 RECNUM FIXED BIN(15,0),
                3 DISPLACE FIXED BIN(15,0),
                3 NAME CHAR(24),
              2 FILESIZE FIXED BIN(15,0),
              2 REC(601),
                3 RECTYPE FIXED BIN(15,0),
                3 RECSIZE FIXED BIN(15,0),
                3 LOCATION FIXED BIN(15,0);
    DCL OTHER CHAR(5440);
    DCL I FIXED BIN(15,0);
    DCL (CKEY,OKEY) FIXED BIN(10,0);
    DCL MET FILE RECORD DIRECT KEYED ENV(REGIONAL(1)) OUTPUT;
    DCL P1 POINTER;
    DCL O CHAR(8) BASED(P1);
    DCL P2 CHAR(2);
    DCL A BIT(64) BASED(P1);
     OPEN FILE(MET) ;
     ALLOCATE O SET(P1);
      CONTROL.FILEID='CLIMAT';
     CONTROL.NUMPASS=0;
     CONTROL.NUMCODE=0;
     CONTROL.NUMLEV=4;
     CONTROL.NUMONE=0;
     CONTROL.FILESIZE=75;    /*  CANADA & RUSSIA COMBINED  */
```

```
      DO I = 1 TO 8;
         CONTROL.PASS(I)=' ';
      END;
      DO I = 1 TO 8;
         CONTROL.LEVNAME(I)=' ';
      END;
      CONTROL.LEVNAME(1)='COUNTRY';
      CONTROL.LEVNAME(2)='REGION';
      CONTROL.LEVNAME(3)='ZONE';
      CONTROL.LEVNAME(4)='STATE';
      DO I = 1 TO 24;
         CONTROL.ONE(I).CODENUM=0;
         CONTROL.ONE(I).NUMDIRS=0;
         CONTROL.ONE(I).NAME=' ';
         CONTROL.ONE(I).RECNUM=-1;
         CONTROL.ONE(I).DISPLACE=0;
      END;
      DO I = 1 TO 601;
         CONTROL.REC(I).RECTYPE=0;
         CONTROL.REC(I).FREESPACE=8440;
         CONTROL.REC(I).LOCATION=0;
      END;
      DO I = 1 TO 32;
         CONTROL.CODE(I).CODENUM=0;
         CONTROL.CODE(I).UNITNAME=' ';
         CONTROL.CODE(I).UNITNUM=0;
         CONTROL.CODE(I).BASE=0;
         CONTROL.CODE(I).CODENAME=' ';
         CONTROL.CODE(I).SCALE=0;
      END;
      OTHER=' ';
      CKEY=0;
      DO I = 1 TO CONTROL.NUMPASS;
         PUT SKIP EDIT('ENTER PASSWORD :')(A);
         GET EDIT(PW)(A(3));
         O=P;
         P=  6;
         CONTROL.PASS(I)=O;
      END;
      WRITE FILE(MET) FROM(CONTROL) KEYFROM(CKEY);
      DO I = 1 TO CONTROL.FILESPECS;
         DKEY=I;
         WRITE FILE(MET) FROM(OTHER) KEYFROM(DKEY);
      END;
      FREE O;
      CLOSE FILE(MET);
      PUT SKIP EDIT('*** END OF JOB ***')(A);
   END INITAL;
```

3.3.2 CONTROL, DIRECTORY, AND DATA DESCRIPTOR ENTRY (YESM001)

YESM001 is used to enter control directory and data descriptive information prior to data entry.

3.3.2.1 Linkages

YESM001 calls YESX002, YESPC01, YESDF01, YESDE01, YESLS01, and YESUD01. YESDE01, YESLS01 and YESUD01 are dummy programs.

3.3.2.2 Interfaces

INITIAL must be run before YESM001.

3.3.2.3 Inputs

See 4.1.2.1.

3.3.2.4 Outputs

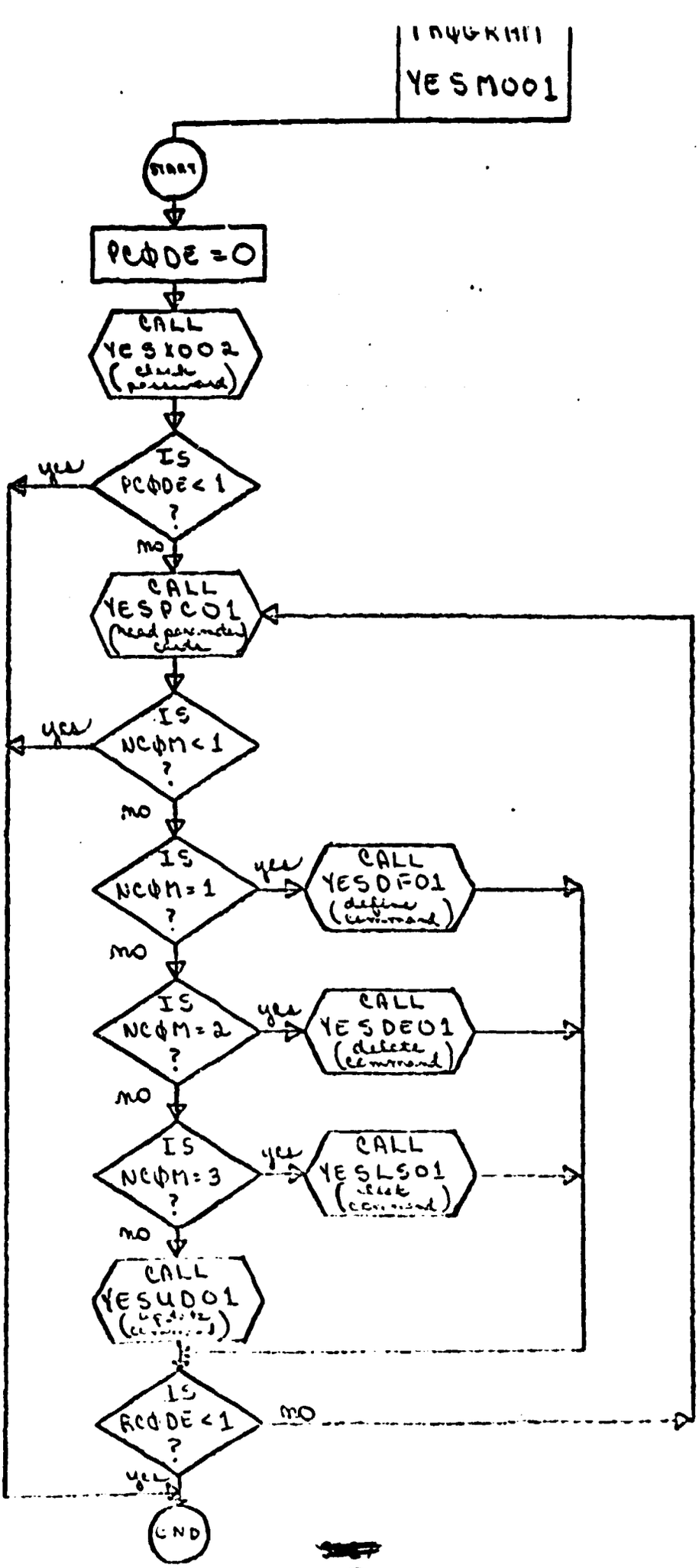
Directory descriptor and control entries in the data base.

3.3.2.5 Flow Chart

Next page.

3.3.2.6 Listing

Follows flow chart.


YESM001
START
PCODE=0
CALL YESK002 (check password)
IS PCODE<1 ?
yes
no
CALL YESPC01 (read parameter file)
IS NCOM<1 ?
yes
no
IS NCOM=1 ?
yes
CALL YESDF01 (define command)
no
IS NCOM=2 ?
yes
CALL YESDE01 (delete command)
no
IS NCOM=3 ?
yes
CALL YESLS01 (list command)
no
CALL YESUD01
IS RCODE<1 ?
no
yes
END

ORIGINAL PAGE IS
OF POOR QUALITY

```
ERON NO. 15      DATE 11/12/75    TIME .0910          LISTING OF MODULE YESM001
DESCRIPTION        DATA BASE PGM
MASTER FILE        W.EDS.CCEA.LEC.LIBR
 ADDED TO MASTER   10/13/75
LAST DATE COPIED     NONE
 LAST UPDATE         NONE

 PASSWORD          GSLX
PROGRAMMER         LEC
 LANGUAGE          PLI
PROC PARAMETER     SNOJCL

  YESM001: PROCEDURE OPTIONS(MAIN);
     DCL YESX002 EXTERNAL ENTRY,
         YESPC01 EXTERNAL ENTRY,
         YESDF01 EXTERNAL ENTRY,
         YESDE01 EXTERNAL ENTRY,
         YESLS01 EXTERNAL ENTRY,
         YESUD01 EXTERNAL ENTRY;
     DCL SYSIN FILE STREAM INPUT;
     DCL SYSPRINT FILE STREAM OUTPUT;
     DCL DAF FILE RECORD DIRECT KEYED ENV(REGIONAL(1));
     DCL PARMS(16) FIXED BIN(15,0);
     DCL (NFILE,NJOB,PCODE,RCODE,NCOM,NOPER,NPARM) FIXED BIN(15,0);
     DCL (ZFLAG,AFLAG) BIT(1);
      OPEN FILE(SYSIN), FILE(SYSPRINT), FILE(DAF) UPDATE;
      NFILE=2;
      NJOB=1;
      ZFLAG='1'B;
      DO WHILE(ZFLAG);
        PCODE=0;
        CALL YESX002(SYSIN,SYSPRINT,DAF,NJOB,PCODE);
        IF PCODE < 1 THEN GOTO EXIT;
        AFLAG='1'B;
        DO WHILE(AFLAG);
          CALL YESPC01(SYSIN,SYSPRINT,NJOB,NCOM,NOPER,NPARM,PARMS);
          IF NCOM < 1 THEN GOTO EXIT;
          ELSE DO;
            RCODE=0;
            IF NCOM = 1 THEN CALL YESDF01(SYSIN,SYSPRINT,DAF,NJOB,RCODE,
              NOPER,NPARM,PARMS);
            ELSE IF NCOM = 2 THEN CALL YESDE01(SYSIN,SYSPRINT,DAF,NJOB,
              RCODE,NOPER,NPARM,PARMS);
            ELSE IF NCOM = 3 THEN CALL YESLS01(SYSIN,SYSPRINT,DAF,NJOB,
              RCODE,NOPER,NPARM,PARMS);
            ELSE CALL YESUD01(SYSIN,SYSPRINT,DAF,NJOB,RCODE,NOPER,NPARM,
              PARMS);
            IF RCODE < 0 THEN GOTO EXIT;
          END;
        END;
      END;
  EXIT: PUT PAGE FILE(SYSPRINT) EDIT('****** END OF PROGRAM ******')(A);
      CLOSE FILE(SYSIN), FILE(SYSPRINT), FILE(DAF);
      RETURN;
    END YESM001;
```

3.3.3 PASSWORD VALIDATION SUBROUTINE (YESX002)

YESX002 is a subroutine called by YESMOOl to validate the users password.

3.3.3.1 Linkages

None.

3.3.3.2 Interfaces

YESX002 searches the password section of the control block.

3.3.3.3 Inputs

Card containing password.

3.3.3.4 Outputs

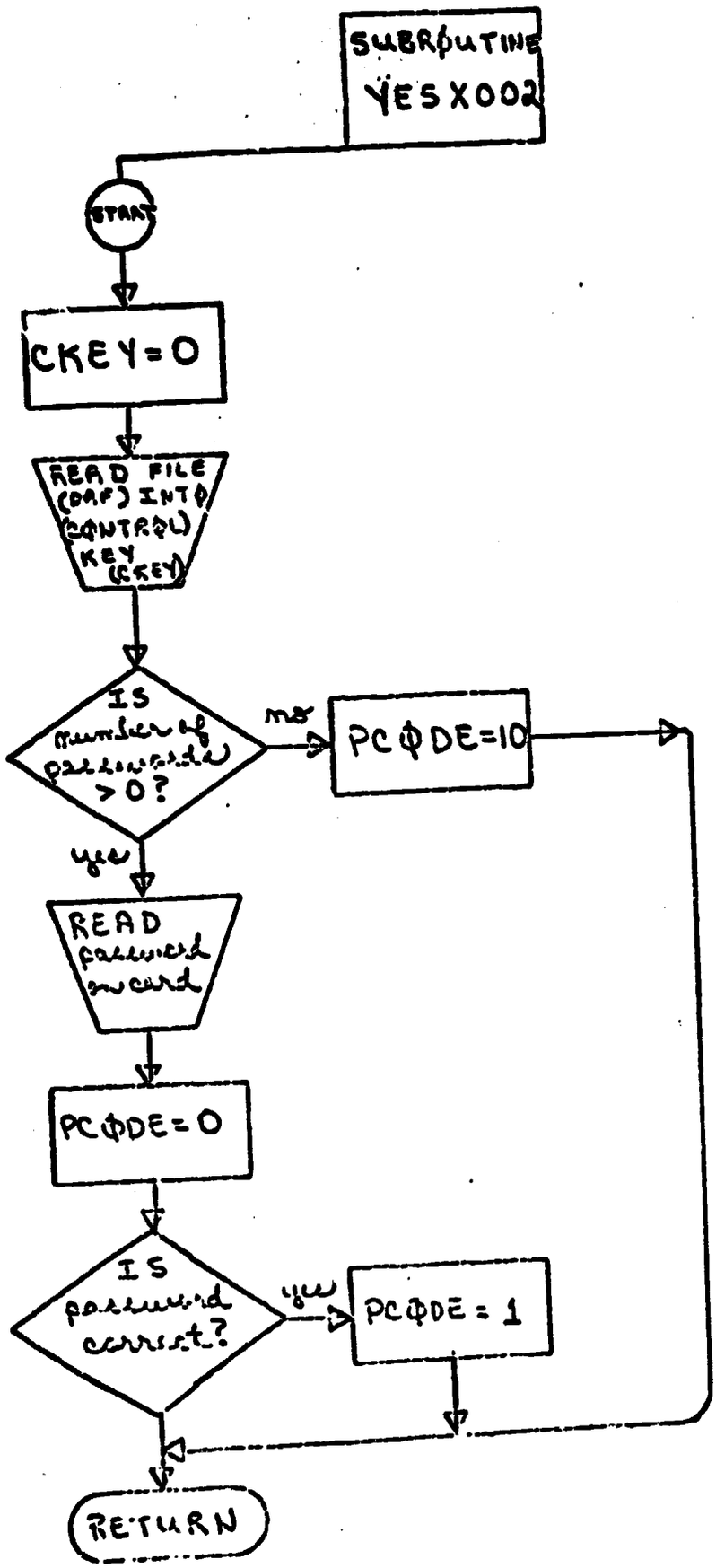
Code allowing or disallowing data base access.

3.3.3.5 Flow Chart

Next page.

3.3.3.6 Listing

Follows flow chart.


SUBROUTINE
YESX002
START
CKEY=0
READ FILE (DAF) INTO (CONTROL) KEY (KEY)
IS number of password > 0?
no
PCODE=10
yes
READ password on card
PCODE=0
IS password correct?
yes
PCODE = 1
RETURN

```
RUN NO. 15      DATE 11/12/76    TIME 0910          LISTIN OF MODULE YESX002
DESCRIPTION     DATA BASE PGM

MASTER FILE     W.EDS.CCEA.LEC.LIBR
 ADDED TO MASTER  10/13/75
LAST DATE COPIED   NONE
LAST UPDATE        NONE

PASSWORD        CONC
PROGRAMMER      LEC
LANGUAGE        PLI
PROC PARAMETER  SNOJCL

YESX002: PROCEDURE(SYSIN,SYSPRINT,DAF,NJOB,PCODE);
   DCL P100 POINTER;
   DCL INA CHAR(80);
   DCL D CHAR(8) BASED(P100);
   DCL M BIT(448) BASED(P100);
   DCL (I,J,PCODE,NJOB) FIXED BIN(15,0);
   DCL CKEY FIXED BIN(15,0);
   DCL (PFLAG,QFLAG) BIT(1);
   DCL (SYSIN,SYSPRINT,DAF) FILE;
   DCL 1 CONTROL,
         2 FILLER CHAR(4),
         2 NUMPASS FIXED BIN(15,0),
         2 PASS(5) CHAR(4),
         2 FILLER CHAR(6365);
   CKEY=0;
   READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
   IF CONTROL.NUMPASS > 0 THEN DO;
     ALLOCATE D SET(P100);
       GET FILE(SYSIN) EDIT(INA)(COL(1),A(80));
       D=SUBSTR(INA,3,8);
       M= '0'B;
       PCODE=0;
       QFLAG='1'B;
       DO J = 1 TO CONTROL.NUMPASS WHILE(QFLAG);
         IF D = CONTROL.PASS(J) THEN DO;
           PCODE=J;
           QFLAG='0'B;
         END;
       END;
       IF PCODE < 1 THEN PUT SKIP FILE(SYSPRINT) EDIT
         ('---- INVALID PASSWORD ----')(A);
     FREE D;
   END;
   ELSE PCODE=10;
   RETURN;
END YESX002;
```

3.3.4 COMMAND CARD DECODING (YESPC01)

YESPC01 is called by YESM001 to decode a command card.

3.3.4.1 Linkages

None.

3.3.4.2 Interfaces

None.

3.3.4.3 Inputs

A command card (see section 4).

3.3.4.4 Outputs

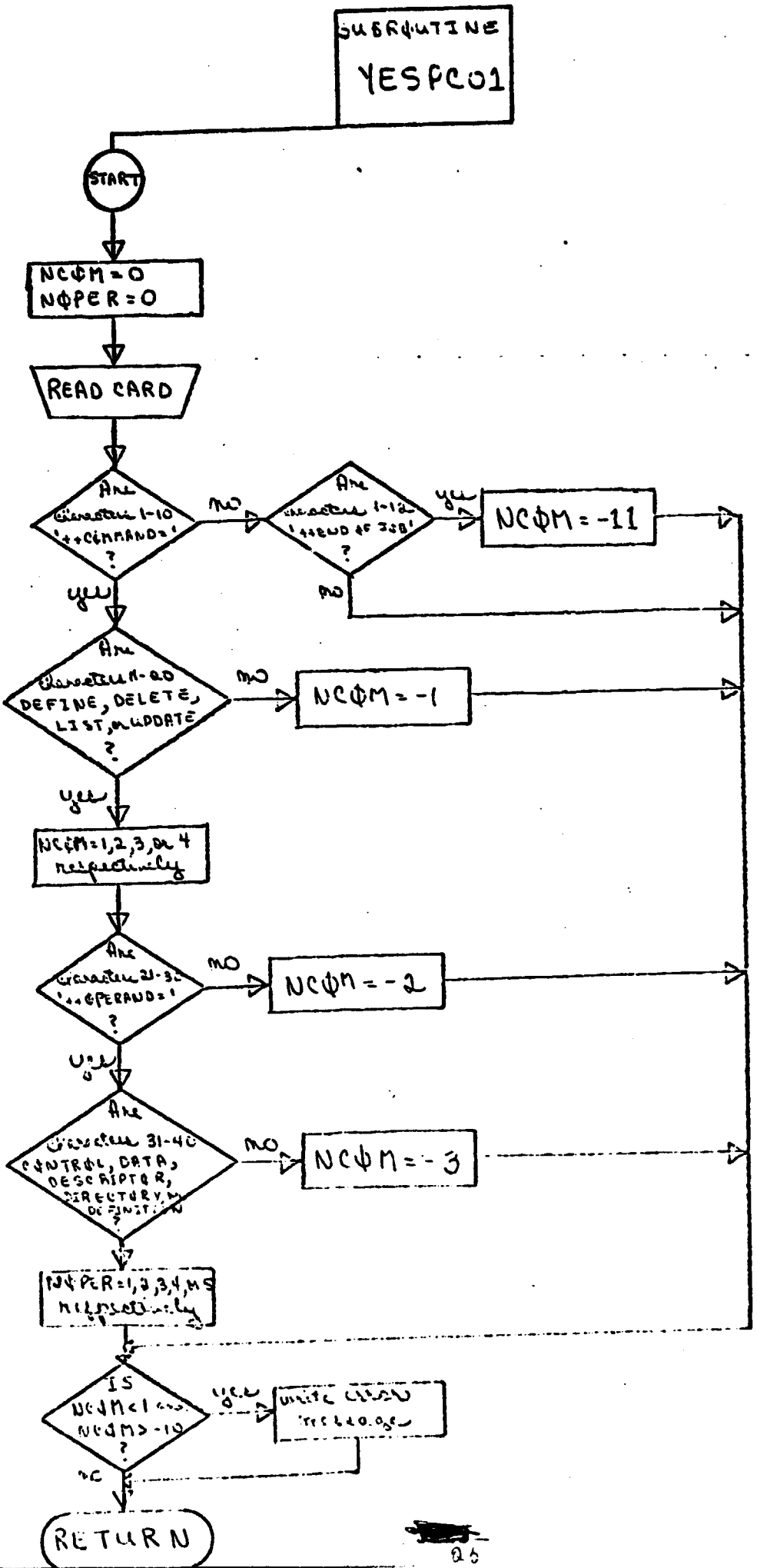
The card is parsed and results returned to YESM001.

3.3.4.5 Flow Chart

Next page.

3.3.4.6 Listing

Follows flow chart.


SUBROUTINE
YESPC01
START
NCΦM = 0
NΦPER = 0
READ CARD
Are Character 1-10 '++COMMAND=' ?
NO
Are Character 1-12 '++END OF JOB' ?
YES
NCΦM = -11
NO
YES
Are Character 1-10 DEFINE, DELETE, LIST, or UPDATE ?
NO
NCΦM = -1
YES
NCΦM = 1, 2, 3, or 4 respectively
Are Character 21-30 '++OPERAND=' ?
NO
NCΦM = -2
YES
Are Character 31-40 CONTROL, DATA, DESCRIPTOR, DIRECTORY, or DEFINITION ?
NO
NCΦM = -3
NΦPER = 1, 2, 3, 4, or 5 respectively
IS NCΦM < 1 and NCΦM > -10 ?
YES
Write Error Message
NO
RETURN

ORIGINAL PAGE IS
OF POOR QUALITY

```
ERON NO. 15      DATE 11/12/76     TIME 0910          LISTING OF MODULE YESPC01
EDESCRIPTION      DATA BASE PGM
EMASTER FILE      W.EDS.CCEA.LEC.LIBR
 ADDED TO MASTER  10/13/76
ELAST DATE COPIED   NONE
 LAST UPDATE        NONE
E
 PASSWORD          XSCR
EPROGRAMMER        LEC
 LANGUAGE          PLI
EPROC PARAMETER    $NOJCL

YESPC01: PROCEDURE(SYSIN,SYSPRINT,NJOB,NCOM,NOPER,NPARM,PARMS);
    DCL INSTR CHAR(80) VARYING;
    DCL (BSTR,CEPR) CHAR(80);
    DCL CCOM CHAR(10);
    DCL COMM(4) CHAR(10) INIT('DEFINE    ','DELETE    ','LIST      ',
        'UPDATE    ');
    DCL OPER(5) CHAR(10) INIT('CONTROL   ','DATA      ','DESCRIPTOR',
    'DIRECTORY ','DEFINITION');
    DCL (NJOB,NCOM,NOPER,NPARM,I,L,F804,F807,F810) FIXED BIN(15,0);
    DCL COFLAG(4,6) BIT(1) INIT('1'B,'0'B,'1'B,'1'B,'1'B,'1'B,
                                '1'B,'0'B,'1'B,'1'B,'1'B,'1'B,
                                '1'B,'1'B,'1'B,'1'B,'1'B,'1'B,
                                '1'B,'1'B,'1'B,'1'B,'1'B,'1'B);
    DCL OPFLAG(4,6) BIT(1) INIT('0'B,'1'B,'1'B,'1'B,'1'B,'0'B,
                                '1'B,'0'B,'1'B,'1'B,'1'B,'0'B,
                                '1'B,'1'B,'1'B,'1'B,'1'B,'0'B,
                                '1'B,'1'B,'1'B,'1'B,'1'B,'0'B);
    DCL (AFLAG,BFLAG) BIT(1);
    DCL (SUBSTR,ONSOURCE) BUILTIN;
    DCL (SYSIN,SYSPRINT) FILE;
    ON CONVERSION BEGIN;
       NCOM=-6;
       GOTO STOP;
    END;
    ON ERROR BEGIN;
      NCOM=-7;
      GOTO STOP;
    END;
    ON ENDFILE(SYSIN) BEGIN;
      NCOM=-9;
      GOTO STOP;
    END;
     NCOM,NOPER,NPARM=0;
     PARMS=0;
       GET FILE(SYSIN) EDIT(BSTR)(COL(1),A(80));
       IF SUBSTR(BSTR,1,10) = '++COMMAND=' THEN DO;
          CCOM=SUBSTR(BSTR,11,10);
          AFLAG='1'B;
          DO I = 1 TO 4 WHILE(AFLAG);
             IF CCOM = COMM(I) THEN DO;
                AFLAG='0'B;
                NCOM=I;
             END;
```

```
        END;
        IF NCOM > 0 THEN DO;
          IF SUBSTR(BSTR,21,10) = '++OPERAND=' THEN DO;
            CCOM=SUBSTR(BSTR,31,10);
            AFLAG='1'B;
            DO I = 1 TO 5 WHILE(AFLAG);
              IF CCOM = OPER(I) THEN DO;
                AFLAG='0'B;
                NOPER=I;
              END;
            END;
            IF NOPER > 0 & COFLAG(NCOM,NOPER)='0'B THEN NCOM=-3;
          END;
          ELSE NCOM=-2;
        END;
        ELSE NCOM=-1;
      END;
      ELSE IF SUBSTR(BSTR,1,12) = '++END OF JOB' THEN NCOM=-11;
      IF NCOM < 1 & NCOM > -10 THEN PUT PAGE FILE(SYSPRINT) EDIT
      ('--- INVALID ++COMMAND CARD ---',BSTR,'--- ERROR CODE NUMBER',
      NCOM)(A,SKIP,A,SKIP,A,F(5,0));
  STOP: RETURN;
END YESPC01;
```

3.3.5 SELECTION OF TYPE OF DEFINITION (YESDF01)

YESDF01 is called by YESM002 tc select the type of definition to be entered.

3.3.5.1 Linkages

YESDF01 calls YESDF02, YESDF03, YESDF04, and YESDF05. YESDF05 is a dummy subroutine.

3.3.5.2 Interfaces

YESDF01 operates on a code produced by YESPC01.

3.3.5.3 Inputs

See 3.3.5.2.

3.3.5.4 Outputs

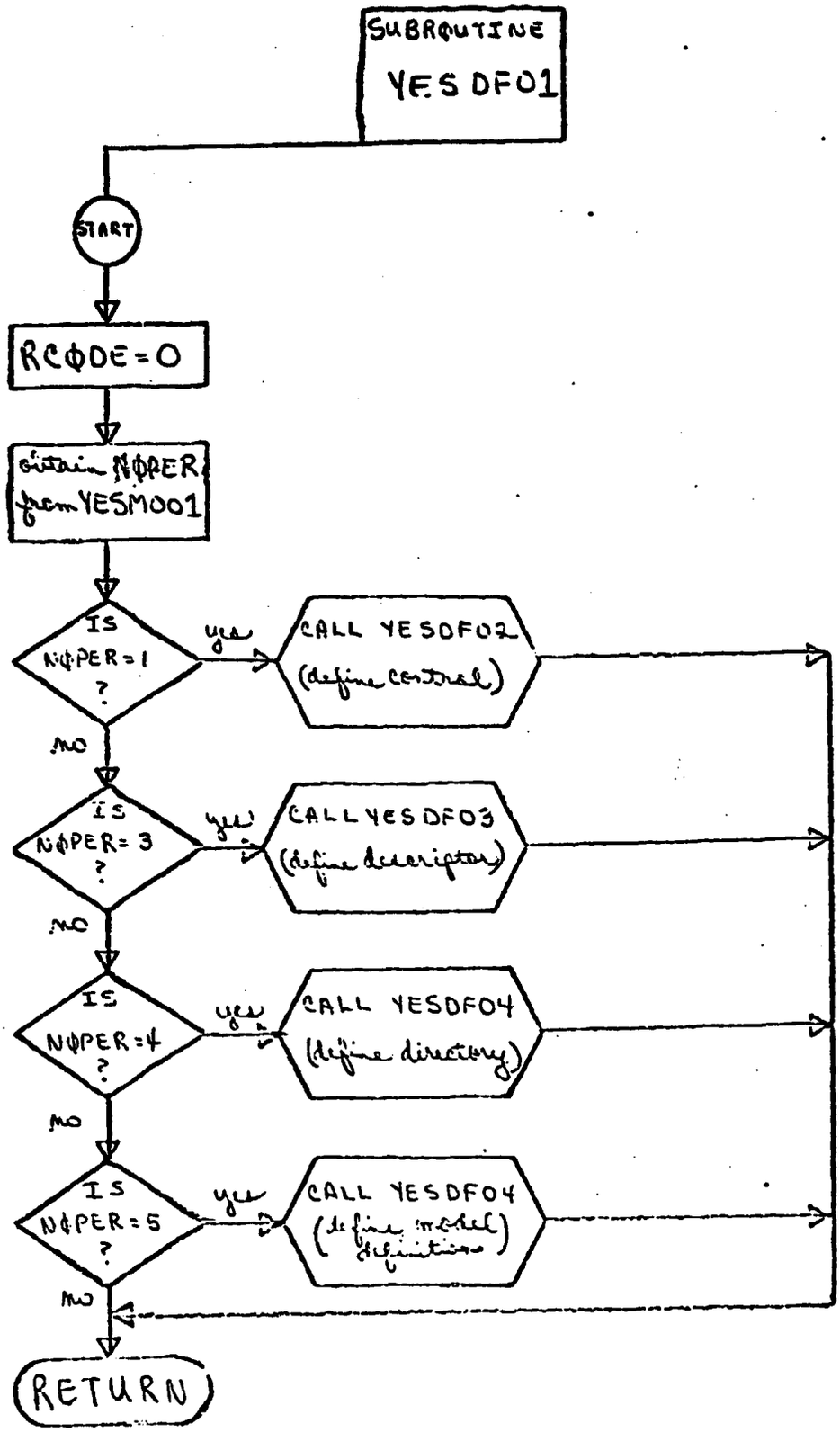
Indirectly – via the called routines.

3.3.5.5 Flow Chart

Next page.

3.3.5.6 Listing

Follows flow chart.


SUBROUTINE
YESDF01
START
RCODE=0
obtain NOPER from YESM001
IS NOPER=1 ?
yes
CALL YESDF02 (define control)
no
IS NOPER=3 ?
yes
CALL YESDF03 (define descriptor)
no
IS NOPER=4 ?
yes
CALL YESDF04 (define directory)
no
IS NOPER=5 ?
yes
CALL YESDF04 (define model definitions)
no
RETURN

ORIGINAL PAGE IS
OF POOR QUALITY

```
CRUN NO. 15      DATE 11/12/76    TIME 0910          LISTING OF MODULE YESDF01
!DESCRIPTION      DATA BASE PGM
!MASTER FILE        W.EDS.CCEA.LEC.LIBR
 ADDED TO MASTER    10/13/76
!LAST DATE COPIED     NONE
 LAST UPDATE          NONE
 PASSWORD           GOVC
!PROGRAMMER         LEC
 LANGUAGE           PLI
!PROC PARAMETER     $NOJCL

  YESDF01: PROC(SYSIN,SYSPRINT,DAF,NJOB,RCODE,NOPER,NPARM,PARMS);
      DCL (SYSIN,SYSPRINT,DAF) FILE;
      DCL (YESDF02,YESDF03,YESDF04,YESDF05) EXTERNAL ENTRY;
      DCL PARMS(16) FIXED BIN(15,0);
      DCL (NJOB,RCODE,NOPER,NPARM) FIXED BIN(15,0);
      RCODE = 0;
      PUT SKIP FILE(SYSPRINT) EDIT('*** DEFINE COMMAND ***')(A);
      IF NOPER = 1 THEN CALL YESDF02(SYSIN,SYSPRINT,DAF,NJOB,RCODE);
      ELSE IF NOPER = 3 THEN CALL YESDF03(SYSIN,SYSPRINT,DAF,NJOB,
        RCODE,NPARM,PARMS);
      ELSE IF NOPER = 4 THEN CALL YESDF04(SYSIN,SYSPRINT,DAF,NJOB,RCODE);
      ELSE IF NOPER = 5 THEN CALL YESDF05(SYSIN,SYSPRINT,DAF,NJOB,
        RCODE,NPARM,PARMS);
      RETURN;
  END YESDF01;
```

3.3.6 CONTROL BLOCK DEFINITION PROGRAM (YESDF02)

YESDF02 enters control block information in the data base.

3.3.6.1 Linkages

None.

3.3.6.2 Interfaces

None.

3.3.6.3 Inputs

Control block definition cards.

3.3.6.4 Outputs

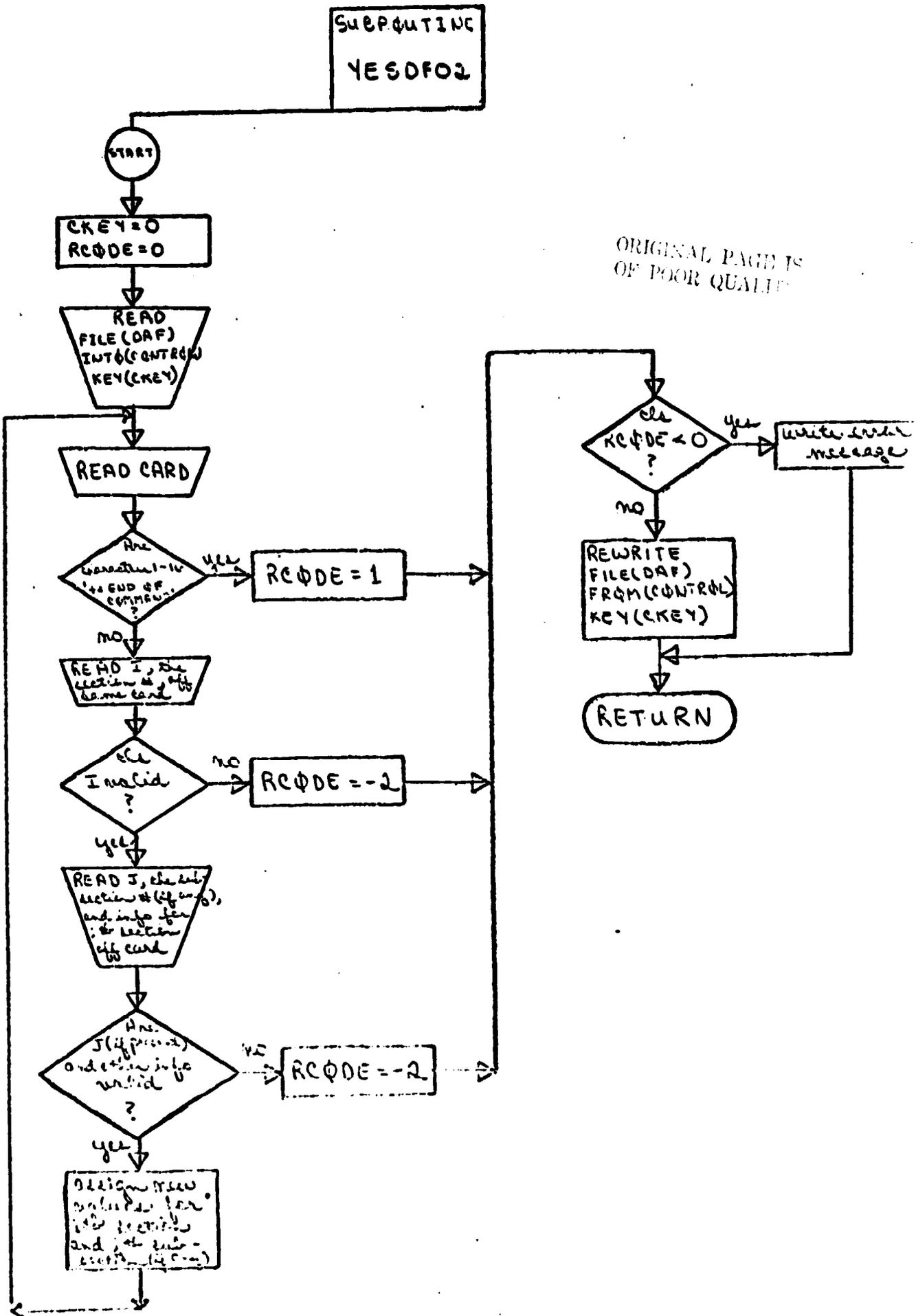
Defined control block to data base.

3.3.6.5 Flow Chart

Next page.

3.3.6.6 Listing

Follows flow chart.



```
RUN NO. 15      DATE 11/12/76   TIME 0918        LISTING OF MODULE YESDF02
DESCRIPTION     DATA BASE PGM

MASTER FILE          *.EDS.CCFA.LEC.LIBR
ADDED TO MASTER      10/13/76
LAST DATE COPIED     NONE
LAST UPDATE          NONE

PASSWORD             XPCH
PROGRAMMER           LEC
LANGUAGE             PLI
PROC PARAMETER       SNOJCL

YESDF02: PROCEDURE(SYSIN,SYSPRINT,DAF,NJOB,RCODE);
/*
/*THIS PROGRAM IS CALLED BY YESDF01 TO DEFINE THE CONTROL BLOCK */
/*
   DCL 1 CONTROL,
         2 FILEID CHAR(8),
         2 NUMPASS FIXED BIN(15,0),
         2 PASS(8) CHAR (8),
         2 NUMLEV FIXED BIN(15,0),
         2 LEVNAME(8) CHAR(24),
         2 NUMCODE FIXED BIN(15,0),
         2 CODE(32),
           3 CODENUM FIXED BIN(15,0),
           3 UNITNUM FIXED BIN(15,0),
           3 BASE FIXED BIN(15,0),
           3 SCALE FIXED BIN(15,0),
           3 CODENAME CHAR(24),
           3 UNITNAME CHAR(24),
         2 NUMONE FIXED BIN(15,0),
         2 ONE(24),
           3 CODENUM FIXED BIN(15,0),
           3 NUMOCRS FIXED BIN(15,0),
           3 RECNUM FIXED BIN(15,0),
           3 DISPLACE FIXED BIN(15,0),
           3 NAME CHAR(24),
         2 FILERECS FIXED BIN(15,0),
         2 REC(40),
           3 RECTYPE FIXED BIN(15,0),
           3 FREESPACE FIXED BIN(15,0),
           3 LOCATION FIXED BIN(15,0);
   DCL CKEY FIXED BIN(10,0);
   DCL P1 POINTER;
   DCL D CHAR(8) BASED(P1);
   DCL A BIT(64) BASED(P1);
   DCL AFLAG BIT(1);
   DCL (I,J,NJOB,RCODE,XNPASS,XNLEV,XNCODE,
     CNO,UNO,XBASE,XSCALE,XONE,CINO,UNO,RECNO,
     XDISP,NT,FS,LOC,FR) FIXED BIN(15,0);
   DCL (XFLEID,PW) CHAR(8);
   DCL (XLNAM,CNAM,UNAM,XNAM) CHAR(24);
   DCL INSTR CHAR(80);
   DCL (SYSIN,SYSPRINT,DAF) FILE;
   DCL SUBSTR BUILTIN;
   PUT SKIP FILE(SYSPRINT) EDIT ('***DEFINE CONTROL PROGRAM***') (A);
   ON CONVERSION BEGIN;
     PUT SKIP FILE(SYSPRINT) EDIT ('***INVALID INPUT CARD***') (A);
     PUT SKIP FILE(SYSPRINT) EDIT (INSTR) (A(80));
     RCODE=-1;
     GOTO STOP;
   END;
   ON ENDFILE(SYSIN) BEGIN;
     PUT SKIP FILE (SYSPRINT) EDIT
       ('***ERROR - END OF FILE SYSIN ENCOUNTERED***') (A);
       RCODE=-1;
       GOTO STOP;
   END;
   CKEY=0;
    READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
      RCODE=0;
      AFLAG='1'B;
```

```
      DO WHILE(AFLAG);
        GET FILE(SYSIN) EDIT(INSTR) (COL(1),A(80));
        IF SUBSTR(INSTR,1,15)='**END OF COMMAND' THEN DO;
          AFLAG='0'B;
          RCODE=1;
        END;
        ELSE DO;
          GET STRING(INSTR) EDIT (I) (X(5),F(2,0));
          IF I<0 | I>11 THEN DO;
            PUT SKIP FILE(SYSPRINT) EDIT ('****',I,
              ' IS AN INVALID SECTION NUMBER****')(A,F(4,0),A);
            AFLAG='0'B; RCODE=-2;
          END;
          ELSE IF I=1 THEN DO;
            GET STRING(INSTR) EDIT(XFLEID) (R(F(12));
            CONTROL.FILEID=XFLEID;
          END;
          ELSE IF I=2 THEN DO;
            GET STRING(INSTR) EDIT(XNPASS) (R(F(1)));
            IF XNPASS<1 | XNPASS>8 THEN DO;
              PUT SKIP FILE(SYSPRINT) EDIT('****',XNPASS,
                'IS AN INVALID NUMBER OF PASSWORDS****')(A,F(4,0),A);
              AFLAG='0'B; RCODE=-2;
            END;
            CONTROL.NUMPASS=XNPASS;
          END;
          ELSE IF I=3 THEN DO;
            ALLOCATE D SET(P);
            GET STRING(INSTR) EDIT(J,PW) (R(F(8)));
            IF J<1 | J>CONTROL.NUMPASS THEN DO;
              IF J<1 THEN PUT SKIP FILE(SYSPRINT) EDIT('****',J,
                'IS AN INVALID NUMBER OF PASSWORDS****')
                (A,F(4,0),A);
              ELSE PUT SKIP FILE(SYSPRINT) EDIT ('****',J,
                'IS LARGER THAN CONTROL.NUMPASS****')
                (A,F(4,0),A);
              AFLAG='0'B; RCODE=-2;
            END;
            D=PW;
            B=-8;
            CONTROL.PASS(J)=D;
          END;
          ELSE IF I=4 THEN DO;
            GET STRING(INSTR) EDIT(XNLEV) (R(F(1)));
            IF XNLEV<1 | XNLEV>4 THEN DO;
              PUT SKIP FILE(SYSPRINT) EDIT('****',XNLEV,
                'IS AN INVALID NUMBER OF LEVELS****')(A,F(4,0),A);
              AFLAG='0'B; RCODE=-2;
            END;
            CONTROL.NUMLEV=XNLEV;
          END;
          ELSE IF I=5 THEN DO;
            GET STRING(INSTR) EDIT(J,XLNAM)(R(F(3,2)));
            IF J<1 | J>CONTROL.NUMLEV THEN DO;
              IF J<1 THEN PUT SKIP FILE(SYSPRINT) EDIT ('****',J,
          'IS AN INVALID NUMBER OF LEVELS****')(A,F(4,0),A);
              ELSE PUT SKIP FILE(SYSPRINT) EDIT('****',J,
                'IS LARGER THAN CONTROL.NUMLEV****')(A,F(4,0),A);
              AFLAG='0'B; RCODE=-2;
            END;
            CONTROL.LEVNAME(J)=XLNAM;
          END;
          ELSE IF I=6 THEN DO;
            GET STRING(INSTR) EDIT(XNCODE) (R(F(1)));
            IF XNCODE<0 | XNCODE>32 THEN DO;
              PUT SKIP FILE(SYSPRINT) EDIT('****',XNCODE,
                'IS AN INVALID NUMBER OF CODES****')(A,F(4,0),A);
              AFLAG='0'B; RCODE=-2;
            END;
            CONTROL.NUMCODE=XNCODE;
          END;
          ELSE IF I=7 THEN DO;
            GET STRING(INSTR) EDIT(J,CNO,U,O,XBASE,XSCALE,
              CNAM,UNAM)(R(F(41)));
            IF J<0 | J>CONTROL.NUMCODE THEN DO;
              IF J<0 THEN PUT SKIP FILE(SYSPRINT) EDIT ('****',J,
                'IS AN INVALID NUMBER OF CODES****')(A,F(4,0),A);
              ELSE PUT SKIP FILE(SYSPRINT) EDIT('****',J,
                'IS LARGER THAN CONTROL.NUMCODE****')(A,F(4,0),A);
              AFLAG='0'B; RCODE=-2;
            END;
```

```
            CONTROL.CODE(J).CODENUM=CNO;
            CONTROL.CODE(J).UNITNUM=UNO;
            CONTROL.CODE(J).BASE=XBASE;
            CONTROL.CODE(J).SCALE=XSCALE;
            CONTROL.CODE(J).CODENAME=CNAM;
            CONTROL.CODE(J).UNITNAME=UNAM;
          END;
          ELSE IF I=8 THEN DO;
            GET STRING(INSTR) EDIT(XONE) (R(F11));
              IF XONE<0 | XONE>24 THEN DO;
                PUT SKIP FILE(SYSPRINT) EDIT('***',XONE,
                  'IS AN INVALID NUMBER OF ONES***')(A,F(4,0),A);
                AFLAG='0'B; RCODE=-2;
              END;
            CONTROL.NUMONE=XONE;
          END;
          ELSE IF I=9 THEN DO;
            GET STRING(INSTR) EDIT(J,CLNO,MOD,RECNO,XDIS,XNAM)
              (R(F42));
              IF J<1 | J>CONTROL.NUMONE THEN DO;
                IF J<1 THEN PUT SKIP FILE(SYSPRINT) EDIT ('***',
                  J,'IS AN INVALID NUMBER OF ONES***')(A,F(4,0),A);
                ELSE PUT SKIP FILE(SYSPRINT) EDIT('***',J,
                  'IS LARGER THAN CONTROL.NUMONE***')(A,F(4,0),A);
                AFLAG='0'B; RCODE=-2;
              END;
            CONTROL.ONE(J).CODENUM=CLNO;
            CONTROL.ONE(J).MODIRS=MOD;
            CONTROL.ONE(J).RECNUM=RECNO;
            CONTROL.ONE(J).DISPLACE=XDIS;
            CONTROL.ONE(J).NAME=XNAM;
          END;
          ELSE IF I=10 THEN DO;
            GET STRING(INSTR) EDIT(FR) (R(F11));
              IF FR<0 | FR>60 THEN DO;
                PUT SKIP FILE(SYSPRINT) EDIT('***',FR,
                  'IS AN INVALID NUMBER OF FILEPECS***')(A,F(4,0),A);
                AFLAG='0'B; RCODE=-2;
              END;
            CONTROL.FILEPECS=FR;
          END;
          ELSE DO;
            GET STRING(INSTR) EDIT(J,PT,FS,LOC)(R(F43));
              IF J<0 | J>CONTROL.FILEPECS THEN DO;
                IF J<0 THEN PUT SKIP FILE(SYSPRINT) EDIT('***',J,
                  'IS AN INVALID NUMBER OF FILEPECS***')(A,F(4,0),A);
                ELSE PUT SKIP FILE(SYSPRINT) EDIT('***',J,
                  'IS LARGER THAN CONTROL.FILEPECS***')(A,F(4,0),A);
                AFLAG='0'B; RCODE=-2;
              END;
            CONTROL.REC(J).RECTYPE=PT;
            CONTROL.REC(J).FRESHNESS=FS;
            CONTROL.REC(J).LOCATION=LOC;
          END;
        END;
      END;
      IF AFLAG='0'B & RCODE<0 THEN PUT SKIP FILE(SYSPRINT)
        EDIT (INSTR) (A);
      IF RCODE<0 THEN GOTO STOP;
      REWRITE FILE(DATA) FROM(CONTROL) KEY(CKEY);
      PUT SKIP FILE(SYSPRINT) EDIT
        ('***CONTROL BLOCK OFFSET 0 UPDATED***') (A);
      F12: FORMAT(X(1),A(8));
      F11: FORMAT(X(1),F(4,0));
      F31: FORMAT(X(7),F(3,0),X(1),A(8));
      F32: FORMAT(X(7),F(3,0),X(1),A(24));
      F41: FORMAT(X(7),F(3,0),4 F(5,0),X(1),A(24),X(1),A(24));
      F42: FORMAT(X(7),F(4,0),4 F(5,0),X(1),A(24));
      F43: FORMAT(X(7),F(3,0),3 F(5,0));
    FREE DO;
  STOP: RETURN;
  END VFSUF02;
```

3.3.7 DATA DESCRIPTOR ENTRY (YESDF03)

YESDF03 enters data descriptors into the data base.

3.3.7.1 Linkages

YESDF03 calls GETDIR.

3.3.7.2 Interfaces

A valid directory entry must exist before invocation.

3.3.7.3 Inputs

Data descriptor cards.

3.3.7.4 Outputs

New data definitions in data base.

3.3.7.5 Flow Chart

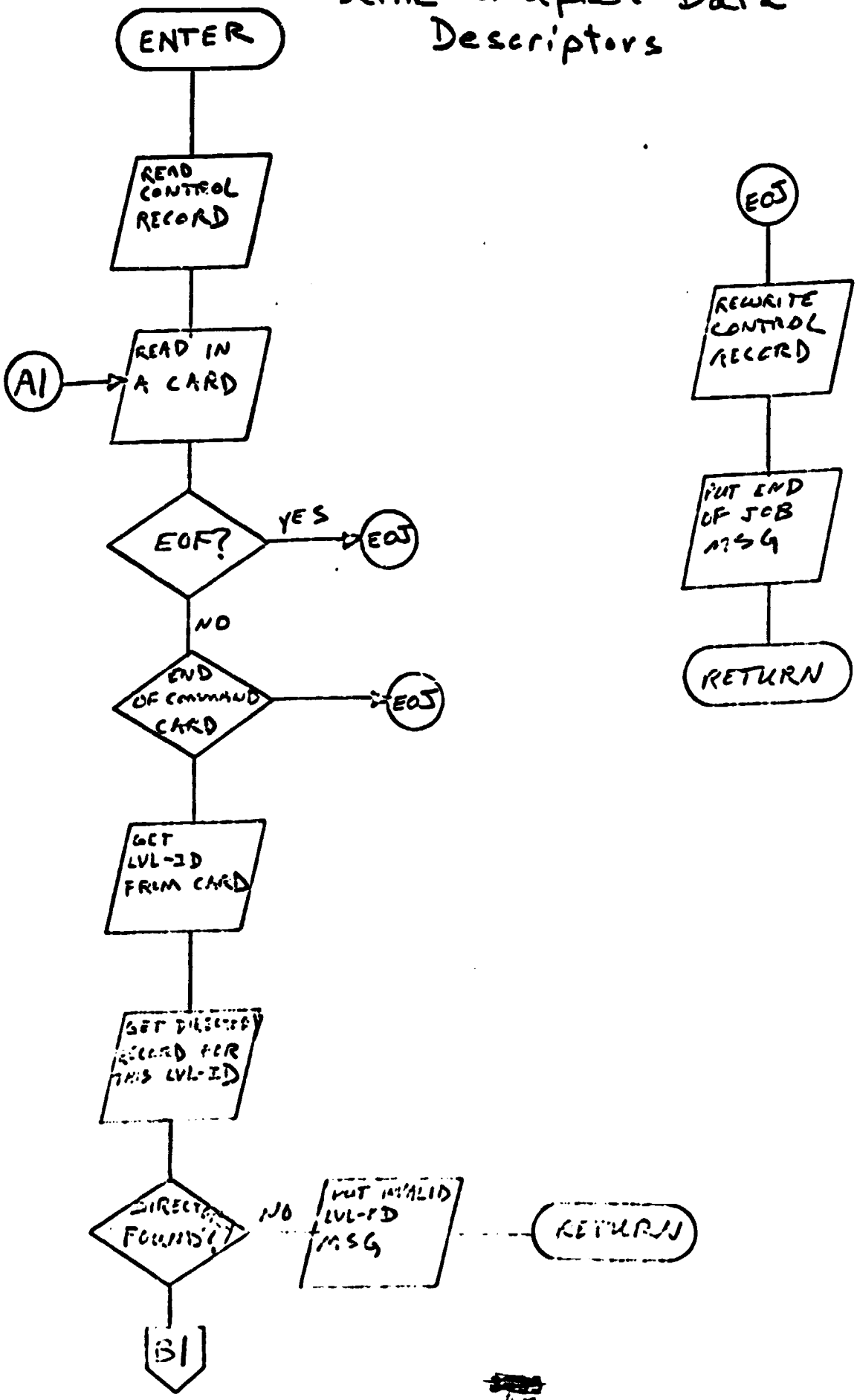
Next page.

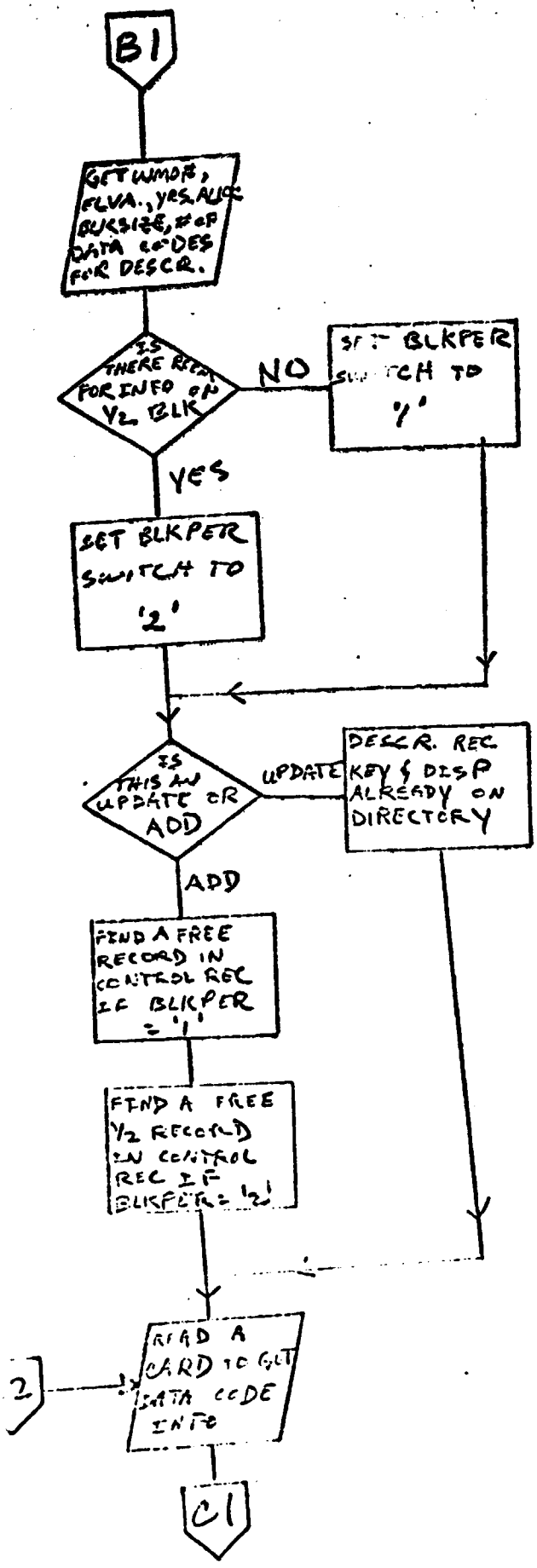
3.3.7.6 Listing

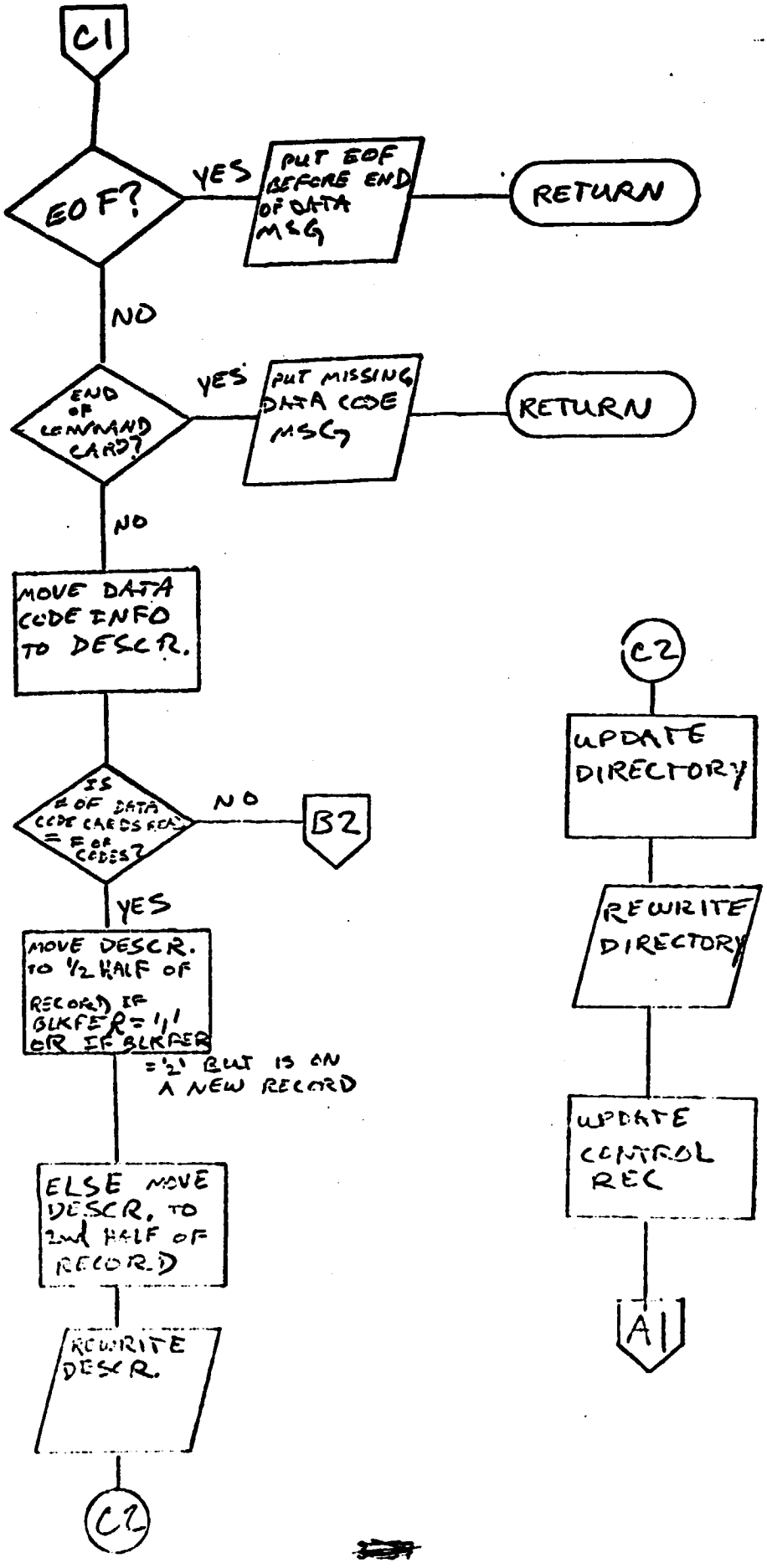
Follows flow chart.

YES DF03

Define or Update Data Descriptors







RUN NO. 15 DATE 11/12/76 TIME 0910 LISTING OF MODULE YESDE03

DESCRIPTION DATA BASE PGM

MASTER FILE W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER 10/13/76
LAST DATE COPIED NONE
LAST UPDATE NONE

PASSWORD RXHC
PROGRAMMER LEC
LANGUAGE PLI
PROC PARAMETER SNOJCL

```
YESDF03: PROC(SYSIN,SYSPRINT,DAF,NJOB,RCODE);                    */
/*  THIS PGM IS CALLED BY YESDF01 TO DEFINE DATA DESCRIPTORS      */
/*                                                                */
    DCL (SYSIN,SYSPRINT,DAF) FILE;
    DCL GETDIR EXTERNAL ENTRY;
    ON ENDFILE(SYSIN) GO TO EOJ;
    DCL LVL_ID_CD PIC '(11)9';
    DCL ENDCHK CHAR(11);
    DCL CKEY FIXED BIN(15) INIT(0);
    DCL PC   FIXED BIN(15) INIT(0);
    DCL DKEY FIXED BIN(15);
    DCL (DDKEY,DIR#) FIXED BIN(15);
    DCL (WMO#,ELEVA,YRS_ALLOC,BLKSZ,NUMCD) FIXED DEC(5,0);
    DCL (CD#,NUM_ELMT,ELSZ,NUMCDS#,SUBCD) FIXED DEC(5,0);
    DCL (NJOB,RCODE) FIXED BIN(15);
    DCL REC_BLK CHAR(6440);
    DCL DIR_CD FIXED BIN(31);
  DCL 1 CONTROL,
      2 FILEID CHAR(8),
      2 NUMPASS FIXED BIN(15,0),
      2 PASS(8) CHAR(8),
      2 NUMLEV FIXED BIN(15,0),
      2 LEVNAME(8) CHAR(24),
      2 NUMCODE FIXED BIN(15,0),
      2 CODE(32),
        3 CODENUM FIXED BIN(15,0),
        3 UNITNUM FIXED BIN(15,0),
        3 BASE FIXED BIN(15,0),
        3 SCALE FIXED BIN(15,0),
        3 CODENAME CHAR(24),
        3 UNITNAME CHAR(24),
      2 NUMONE FIXED BIN(15,0),
      2 ONE(24),
        3 CODENUM FIXED BIN(15,0),
        3 NUMDIRS FIXED BIN(15,0),
        3 RECNUM FIXED BIN(15,0),
        3 DISPLACE FIXED BIN(15,0),
        3 NAME CHAR(24),
      2 FILERECS FIXED BIN(15,0),
      2 REC(601),
        3 RECTYPE FIXED BIN(15,0),
        3 FREESPACE FIXED BIN(15,0),
        3 LOCATION FIXED BIN(15,0);
  DCL 1 DIR#,
      2 DIR (64),
        3 LEVNUM FIXED BIN(15,0),
        3 CODENUM# FIXED BIN(15,0),
        3 LAT FIXED BIN(15,0),
        3 LON FIXED BIN(15,0),
        3 DIRNAME CHAR(24),
        3 PREC FIXED BIN(15,0),
        3 PDISP FIXED BIN(15,0),
        3 BREC FIXED BIN(15,0),
        3 BDISP FIXED BIN(15,0),
        3 CREC FIXED BIN(15,0),
        3 CDISP FIXED BIN(15,0),
        3 DREC FIXED BIN(15,0),
        3 DDISP FIXED BIN(15,0),
        3 LEVCODE FIXED BIN(31,0),
```

```
          3 MODEL(4),
            4 CROP FIXED BIN(15,0),
            4 MREC FIXED BIN(15,0),
          4 MDISP FIXED BIN(15,0),
       2 FILLER CHAR (56);
     DCL 1 DATADESC,
           2 ID                    FIXED BIN(31,0),
           2 WMO                   FIXED BIN(31,0),
           2 LATI                  FIXED BIN(15,0),
           2 LONGI                 FIXED BIN(15,0),
           2 ELEV                  FIXED BIN(15,0),
           2 TOTALBLKS_ALLOC       FIXED BIN(15,0),
           2 NUMRECS_USED          FIXED BIN(15,0),
           2 BLOCKSIZE             FIXED BIN(15,0),
           2 FSTRECNO              FIXED BIN(15,0),
           2 FSTDISP               FIXED BIN(15,0),
           2 LSTRECNO              FIXED BIN(15,0),
           2 LSTDISP               FIXED BIN(15,0),
           2 RESERVED              CHAR(18),
           2 NUMSCODE              FIXED BIN(15,0),
           2 DCODE(12),
             3 CODENUMB              FIXED BIN(15,0),
             3 NUMSELEM              FIXED BIN(15,0),
             3 ELEMSIZE              FIXED BIN(15,0),
             3 NUMSCODE              FIXED BIN(15,0),
             3 SUBCODE(8)            FIXED BIN(15,0),
         2 FILLER2 CHAR(2484);
     DCL 1 DATA_BLK1 BASED(P),
           2 DESC1 LIKE DATADESC,
           2 FIL1  CHAR(3220);
     DCL 1 DATA_BLK2 BASED(P),
           2 FIL2  CHAR(3220),
           2 DESC2 LIKE DATADESC;
     DCL BLKPER CHAR(1) INIT('1');
     DCL LD1    CHAR(1) INIT('1');
     DCL UPDTE_SW  CHAR(1) INIT('0');  /* 0=ADD  1=UPDATE  */
     ALLOCATE DATA_BLK1;
   ON CONVERSION GO TO ERR3;
     RCODE= -1;
     DATADESC.FILLER2 = '    ';
     READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
 NXT_ONE:  ;
     GET SKIP FILE(SYSIN) EDIT(ENDCHK)(A(11));
     IF ENDCHK = '**END OF CD' THEN GO TO EOJ;
     LVL_ID_CD = TRANSLATE(ENDCHK,'0',' ');
     DIR_CD = LVL_ID_CD;
     CALL GETDIP(DIR_CD,DIRX,CONTROL,DAF,DKEY,DIR#,RC);
     IF RC = -1 THEN DO;
       PUT SKIP DATA(LVL_ID_CD,DIR#,RC);
       GO TO ERR1;
       END;
     GET FILE(SYSIN) LIST(WMO#,ELEVA,YRS_ALLOC,BLKSZ,NUMCD);
     IF (YRS_ALLOC * BLKSZ + 306) < 3220 THEN BLKPER = '2';
     ELSE BLKPER = '1';
     IF DREC(DIR#) ¬= -1 THEN DO;
        /*  RECORD ALREADY ASSIGNED  */
        /*  THIS IS AN UPDATE        */
       UPDTE_SW = '1';
       I = DREC(DIR#);
     IF RECTYPE(I) ¬= 1 THEN DO;
       PUT SKIP LIST('***WARNING -- CONTROL BLOCK WAS NOT UPDATED
 THIS UPDATE WILL TRY TO CORRECT THE CONTROL BLOCK ***');
       RECTYPE(I) = 1;
       END;
       ODKEY = I;
       READ FILE(DAF) INTO(DATA_BLK1) KEY(ODKEY);
       IF BLKPER = '1' THEN  DATADESC = DESC1;
         ELSE  IF LD1 = '1' THEN  DATADESC = DESC1;
               ELSE DATADESC = DESC2;
       GO TO EXIT_DO;
       END;
     /*  OTHERWISE                                    */
     /*  LOOK IN CONTROL BLK FOR 1ST AVAILABLE RECORD  */
     DO I = 1 TO FILERECS;
     IF RECTYPE(I) = 1 THEN
       IF BLKPER = '2' THEN
         IF FREESPACE(I) = 3220 THEN GO TO EXIT_DO;
     IF RECTYPE(I) = 0 THEN GO TO EXIT_DO;
     END;
```

```
    /* ERROR */
    PUT SKIP LIST('**ERROR YOU HAVE LOST ALL YOUR MARBLES **');
    RETURN;
    /*      */
EXIT_DO:  ;
    DREC(DIR#)=1;
    IF BLKPER = '1' THEN DDISP(DIR#) = 1;
    ELSE IF LD1 = '1' THEN DDISP(DIR#) = 1;
      ELSE DDISP(DIR#) = 3221;
    DATADESC.ID = DIR_CD;
    DATADESC.WMO = WMO#;
    DATADESC.ELEV = ELEVA;
    DATADESC.TOTALBLKS_ALLOC = YRS_ALLOC;
    DATADESC.BLOCKSIZE = BLKSZ;
    /*  GET INFO FROM DIR FOR DATA DESC  */
    DATADESC.LATI = LAT(DIR#);
    DATADESC.LONGI = LON (DIR#);
    DATADESC.RESERVED = '  ';
    DO J = 1 TO 12;
      DATADESC.DCODE(J).CODE_NUM = 0;
      DATADESC.DCODE(J).NUMSELEM = 0;
      DATADESC.DCODE(J).ELEMSIZE = 0;
      DATADESC.DCODE(J).NUMSCODE = 0;
      DO K = 1 TO 8;
        DATADESC.DCODE(J).SUBCODE(K) = 0;
        END;
    END;
    DATADESC.NUMBCODE = NUMCO;
    DO J = 1 TO DATADESC.NUMBCODE;
    GET SKIP FILE(SYSIN) LIST(CO#,NUM_ELMT,ELSZ,NUMCOSB);
    IF ENDCHK = '++END OF CO' THEN GO TO ERR2;
    DATADESC.DCODE(J).CODE_NUM = CO#;
    DATADESC.NUMSELEM(J) = NUM_ELMT;
    DATADESC.ELEMSIZE(J) = ELSZ;
    DATADESC.NUMSCODE(J) = NUMCOSB;
    IF DATADESC.NUMSCODE(J) > 0 THEN DO;
      DO K = 1 TO DATADESC.NUMSCODE(J);
      GET FILE(SYSIN) LIST(SUBCD);
       IF ENDCHK = '++END OF CO' THEN GO TO ERR2;
       ELSE;
      DATADESC.DCODE(J).SUBCODE(K) = SUBCD;
      END;
                END;
    END;
  IF UPDTE_SW = '1' THEN DO;     /* UPDATE RECORD */
    IF BLKPER = '1' THEN DESC1 = DATADESC;
      ELSE IF LD1 = '1' THEN DESC1 = DATADESC;
        ELSE DESC2 = DATADESC;
    REWRITE FILE(DAF) FROM(DATA_BLK1) KEY(DDKEY);
     UPDTE_SW = '0';
    GO TO UPDATE_DIR;
    END;
  DATADESC.NUMBYRS_USED = 0;
  DATADESC.FSTRECNO = 1;
  IF LD1 = '1' THEN DATADESC.FSTDISP = 337;
  ELSE DATADESC.FSTDISP = 3557;
  DATADESC.LSTRECNO = 1;
  IF LD1 = '1' THEN DATADESC.LSTDISP = 337;
  ELSE DATADESC.LSTDISP = 3557;
  IF BLKPER = '1' THEN
  L = DATADESC.BLOCKSIZE * DATADESC.TOTALBLKS_ALLOC/6440 + .5;
  ELSE
    L = DATADESC.BLOCKSIZE * DATADESC.TOTALBLKS_ALLOC/3220+.5;
  IF L > 1 THEN PUT SKIP FILE(SYSPRINT)
  LIST('** DATA DOESNT FIT FOR 1 RECORD *  L = ',L);
  DDKEY = 1;    /* KEY TO DD RECORD  */
  IF BLKPER = '1' THEN DO;
    DESC1 = DATADESC;
    FIL1 = ' ';
    REWRITE FILE(DAF) FROM(DATA_BLK1) KEY(DDKEY);
    RECTYPE(1) = 1;
    FRESPACE(1) = 6440 - 335 - DESC1.TOTALBLKS_ALLOC *
```

```
                    DESC1:BLOCKSIZE;
        LOCATION(I) = 6440 - FRESPACE(I) + 1;
        END;
      ELSE DO;
        IF LD1 = '1' THEN DO;
        DESC1 = DATADESC;
        FIL1 = ' ';
        REWRITE FILE(DAF) FROM(DATA_BLK1) KEY(DDKEY);
        RECTYPE(I) = 1;
        FRESPACE(I) = 3220;
        LOCATION(I) = 3221;
        LD1 = '2';
        END;
      ELSE DO;
        DESC2 = DATADESC;
        REWRITE FILE(DAF) FROM(DATA_BLK1) KEY(DDKEY);
        RECTYPE(I) = 1;
        FRESPACE(I) = 0;
        LOCATION(I) = -1;
        LD1 = '1';
        END;
      END;
UPDATE_DIR:  ;
      /* REWRITE DIR  */
      REWRITE FILE(DAF) FROM(DIRX) KEY(DKEY);
      GO TO NXT_ONE;
EOJ:  ;
      REWRITE FILE(DAF) FROM (CONTROL) KEY(CKEY);
      PUT SKIP FILE(SYSPRINT) LIST('DATA DESCRIPTORS ADDED TO FILE');
      FREE DATA_BLK1;
    RCODE = 1;
      RETURN;
ERR1:  ;
      PUT SKIP FILE(SYSPRINT) LIST('** INVALID LEVEL CODE **', ENDCHK);
      RETURN;
 ERR2:  ;
      PUT SKIP FILE(SYSPRINT) LIST
      ('MISSING CODE DATA **');
      RETURN;
 ERR3:  ;
      PUT SKIP FILE(SYSPRINT) LIST
      ('** NON-NUMERIC DATA ON INPUT - CORRECT AND RESUBMIT **');
      RETURN;
  END YESDF03;
```

3.3.8 RECOVER DIRECTORY FROM THE DATA BASE

GETDIR is used by YESM001 and the loaders (3.3) to recover directory blocks from the data base.

3.3.8.1 Linkages

None.

3.3.8.2 Interfaces

None.

3.3.8.3 Inputs

Level identification number.

3.3.8.4 Outputs

Directory block, or error indication.

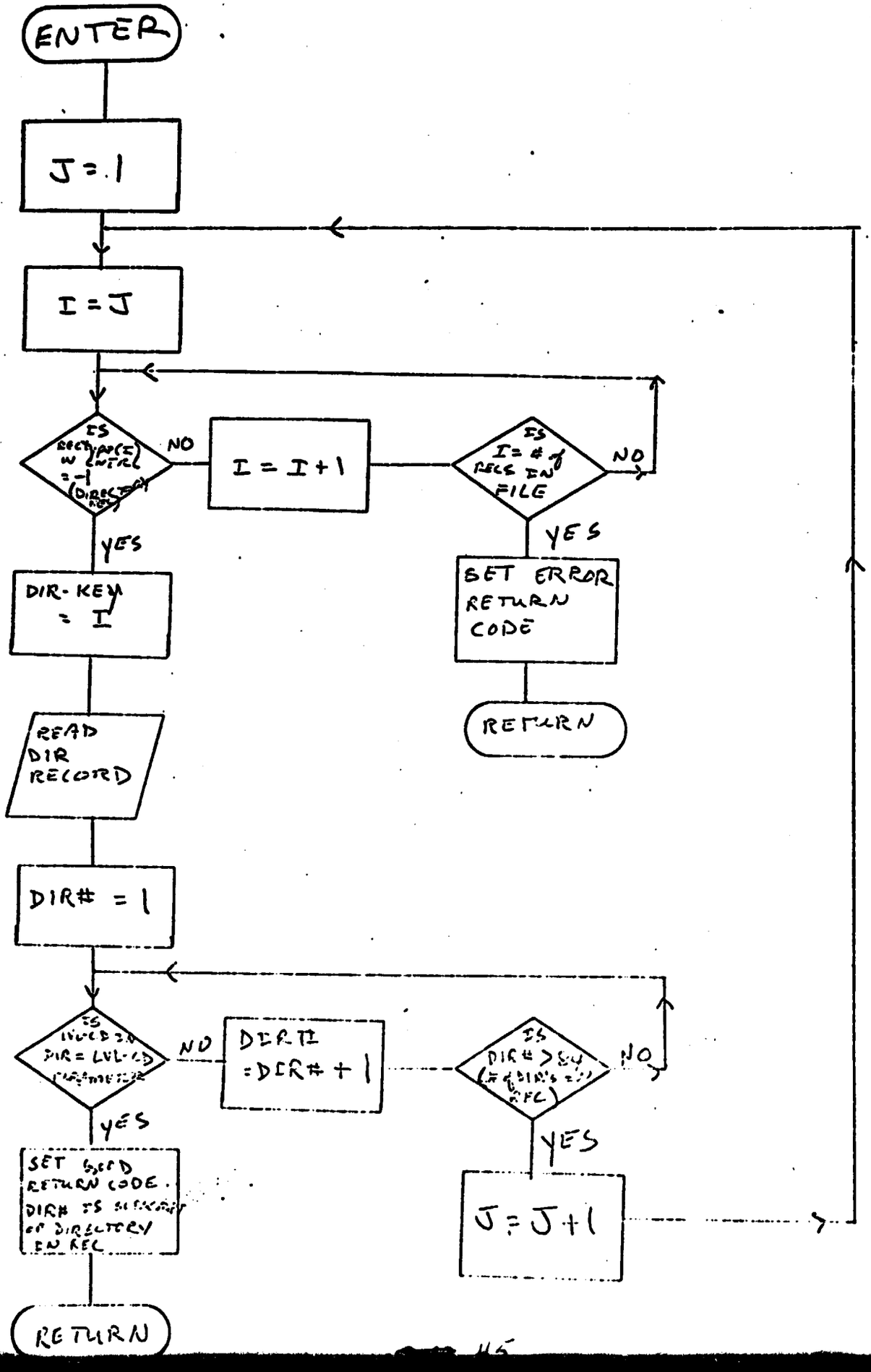
3.3.8.5 Flow Chart

Next page.

3.3.8.6 Listing

Follows flow chart.

GETDIR


ENTER
J = 1
I = J
IS RECTYPE(E) W CNTRL = -1 (DIRECTORY)
NO
I = I + 1
IS I = # of RECS IN FILE
NO
YES
SET ERROR RETURN CODE
RETURN
YES
DIR-KEY = I
READ DIR RECORD
DIR# = 1
IS INVCD IN DIR = LVL-CD
NO
DIR# = DIR# + 1
IS DIR# > (# of DIR's in REC)
NO
YES
J = J + 1
YES
SET GOOD RETURN CODE. DIR# IS SUBSCRIPT OF DIRECTORY IN REC
RETURN

```
RUN NO. 15      DATE 11/12/76    TIME 0910          LISTING OF MODULE GETDIR
DESCRIPTION     DATA BASE PGM
MASTER FILE     W.EOS.CCEA.LEC.LIBR
 ADDED TO MASTER   10/13/76
LAST DATE COPIED    NONE
 LAST UPDATE        NONE
PASSWORD        CHAK
PROGRAMMER      LEC
LANGUAGE        PLI
PROC PARAMETER  $NOJCL
```



```
GETDIR:  PROC(DIR_CD,DIRX,CONTROL,DAF,DKEY,DIR#,RC);
         DCL DAF FILE;
         DCL (DIR#,DKEY,RC) FIXED BIN(15);
       DCL DIR_CD FIXED BIN(31);
      DCL 1 CONTROL,
            2 FILEID CHAR(8),
            2 NUMPASS FIXED BIN(15,0),
            2 PASS(8) CHAR(8),
            2 NUMLEV FIXED BIN(15,0),
            2 LEVNAME(8) CHAR(24),
            2 NUMCODE FIXED BIN(15,0),
            2 CODE(32),
              3 CODENUM FIXED BIN(15,0),
              3 UNITNUM FIXED BIN(15,0),
              3 BASE FIXED BIN(15,0),
              3 SCALE FIXED BIN(15,0),
              3 CODENAME CHAR(24),
              3 UNITNAME CHAR(24),
            2 NUMDIR FIXED BIN(15,0),
            2 DIR(24),
              3 CODENUM FIXED BIN(15,0),
              3 NUMDIRS FIXED BIN(15,0),
              3 RECNUM FIXED BIN(15,0),
              3 DISPLACE FIXED BIN(15,0),
              3 NAME CHAR(24),
            2 FILERECS FIXED BIN(15,0),
            2 REC(60),
              3 RECTYPE FIXED BIN(15,0),
              3 FREESPACE FIXED BIN(15,0),
              3 LOCATION FIXED BIN(15,0);
      DCL 1 DIRX,
            2 DIR (84),
              3 LEVNUM FIXED BIN(15,0),
              3 CODENUM FIXED BIN(15,0),
              3 LAT FIXED BIN(15,0),
              3 LON FIXED BIN(15,0),
              3 DIRNAME CHAR(24),
              3 PREC FIXED BIN(15,0),
              3 PDISP FIXED BIN(15,0),
              3 HREC FIXED BIN(15,0),
              3 HDISP FIXED BIN(15,0),
              3 CREC FIXED BIN(15,0),
              3 CDISP FIXED BIN(15,0),
              3 DREC FIXED BIN(15,0),
              3 DDISP FIXED BIN(15,0),
              3 LEVCODE FIXED BIN(31,0),
              3 MODEL(4),
                4 CREC FIXED BIN(15,0),
                4 MREC FIXED BIN(15,0),
                4 MDISP FIXED BIN(15,0),
            2 FILLER CHAR (52);
       DCL 1 REC_BLK LIKE DIRX;
       ON CONVERSION BEGIN;
       PUT SKIP LIST(DIR_CD,DIR#,RC,DKEY,ONE(1),DIRX,DIR(DIR#));
       END;
/*                    */
       J=1;
  LP1: ;
       DO I=J TO FILERECS;
       IF RECTYPE(I) = -1 THEN GOTO XIT1;
       END;
       RC = -1;   /* ERROR */
       RETURN;
/*    */
  XIT1: ;
       DKEY = 1;
       READ FILE(DAF) INTO(REC_BLK) KEY(DKEY);
       DIRX = REC_BLK;
  LP2: ;
       DO DIR# = 1 TO 24;
       IF DIRX.LEVCODE(DIR#) = DIR_CD THEN GOTO FOUND_DIR;
       END;
       J = J + 1;
       GOTO LP1;
  FOUND_DIR: ;
       RC = 1;
       RETURN;
  END GETDIR;
```

3.3.9 DIRECTORY BLOCK ENTRY ROUTINE (YESDF04)

YESDF04 places directory blocks in the data base.

3.3.9.1 Linkages

None.

3.3.9.2 Interfaces

None.

3.3.9.3 Inputs

Directory definition cards.

3.3.9.4 Outputs

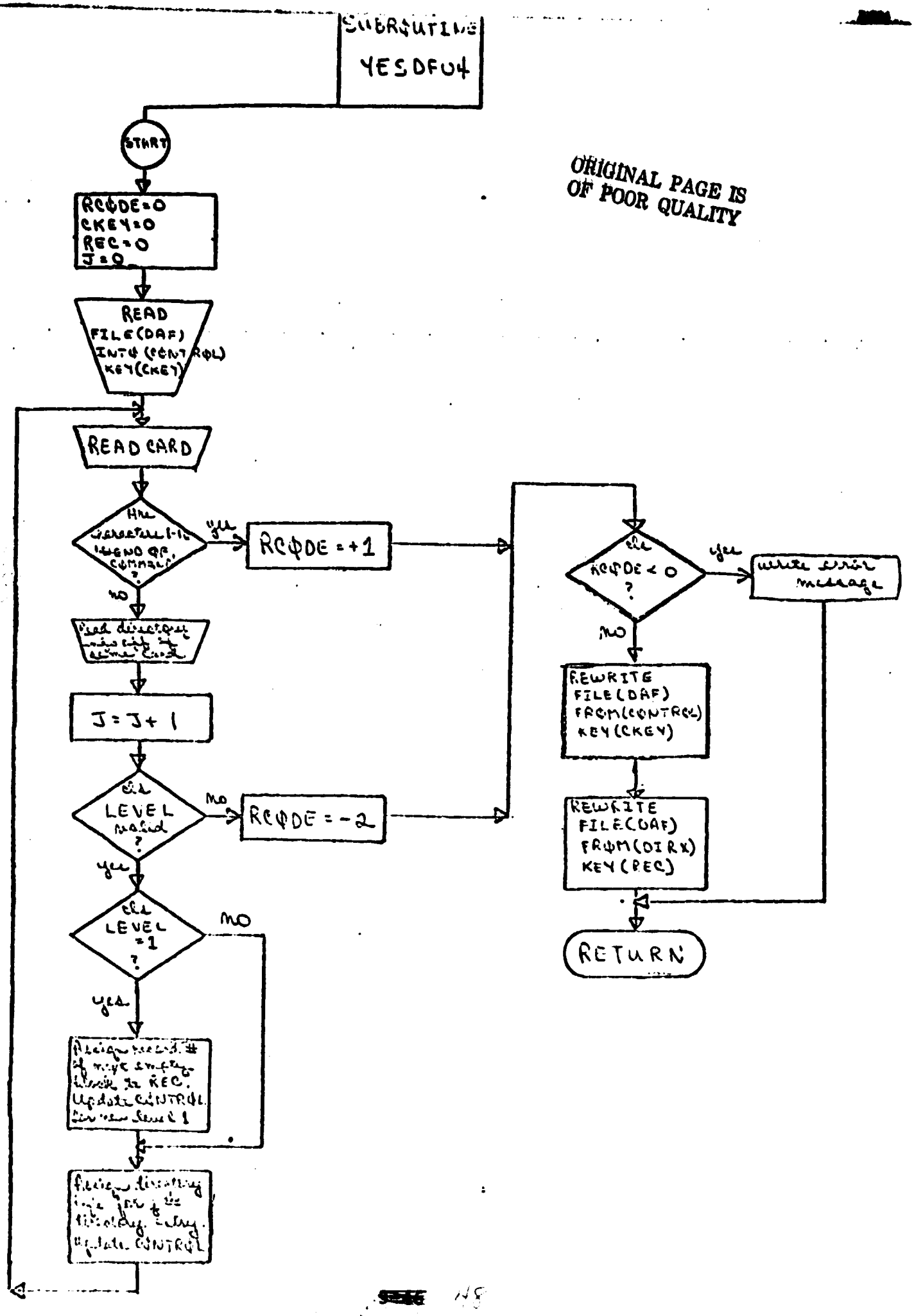
Directory definition on the data base.

3.3.9.5 Flow Chart

Next page.

3.3.9.6 Listing

Follows flow chart.



```
$RUN NO. 15      DATE 11/12/74    TIME 0910          LISTING OF MODULE YESDF04
$DESCRIPTION.    DATA BASE PGM
$MASTER FILE     W.EDS.CCEA.LEC.LIBR
$ADDED TO MASTER 10/13/75
$LAST DATE COPIED   NONE
$LAST UPDATE        NONE
$
$PASSWORD        HOMH
$PROGRAMMER      LEC
$LANGUAGE        PLI
$PROC PARAMETER  $NOJCL

YESDF04: PROCEDURE(SYSIN,SYSPRINT,DAF,JJJ,RCODE);                      */
/* THIS PROGRAM IS CALLED BY YESDF01 TO DEFINE DIRECTORY ENTRIES.      */
/* THIS PROGRAM ASSUMES A 6440-BYTE DIRECTORY BLOCK CAN CONTAIN        */
/*   84 DIRECTORY ENTRIES, EACH 76 BYTES LONG.                         */
   DCL 1 CONTROL,
         2 FILEID CHAR(8),
         2 NUMPASS FIXED BIN(15,0),
         2 PASS(8) CHAR(8),
         2 NUMLEV FIXED BIN(15,0),
         2 LEVNAME(8) CHAR(24),
         2 NUMCODE FIXED BIN(15,0),
         2 CODE(32),
           3 CODENUM FIXED BIN(15,0),
           3 UNITNUM FIXED BIN(15,0),
           3 BASE FIXED BIN(15,0),
           3 SCALE FIXED BIN(15,0),
           3 CODENAME CHAR(24),
           3 UNITNAME CHAR(24),
         2 NUMDIR FIXED BIN(15,0),
         2 ONE(24),
           3 CODENUM FIXED BIN(15,0),
           3 NUMDIRS FIXED BIN(15,0),
           3 RECNUM FIXED BIN(15,0),
           3 DISPLACE FIXED BIN(15,0),
           3 NAME CHAR(24),
         2 FILERECS FIXED BIN(15,0),
         2 REC(201),
           3 RECTYPE FIXED BIN(15,0),
           3 FREESPACE FIXED BIN(15,0),
           3 LOCATION FIXED BIN(15,0);
   DCL 1 DIRX BASED(P),
         2 DIR (84),
           3 LEVNUM FIXED BIN(15,0),
           3 CODENUMB FIXED BIN(15,0),
           3 LAT FIXED BIN(15,0),
           3 LON FIXED BIN(15,0),
           3 DIRNAME CHAR(24),
           3 PREC FIXED BIN(15,0),
           3 PDISP FIXED BIN(15,0),
           3 BREC FIXED BIN(15,0),
           3 BDISP FIXED BIN(15,0),
           3 CREC FIXED BIN(15,0),
           3 CDISP FIXED BIN(15,0),
           3 DREC FIXED BIN(15,0),
           3 DDISP FIXED BIN(15,0),
           3 LEVCODE FIXED BIN(31,0),
           3 MODEL(4),
             4 CROP FIXED BIN(15,0),
             4 MREC FIXED BIN(15,0),
             4 MDISP FIXED BIN(15,0),
         2 FILLER CHAR (20);
   DCL (REC,CKEY,LASTKEY) FIXED BIN(10,0);
   DCL (I,J,K,JJ,LEV,XCODE,XLAT,XLON,XP,XR,XC,MCODE,XSIZE,RTYPE,L,
     II) FIXED BIN(15,0);
   DCL (XL,D) FIXED BIN(31,0);
   DCL INSTR CHAR(80);
   DCL ANAME CHAR(24);
   DCL (AFLAG,BFLAG,CFLAG) BIT(1);
   DCL (SYSIN,SYSPRINT,DAF) FILE;
   DCL SUBSTR BUILTIN;
   DCL BLANK CHAR(6000) BASED(P);
   DCL P POINTER;
   PUT SKIP FILE(SYSPRINT) EDIT ('**DEFINE DIRECTORY PROGRAM**') (A);
   ON CONVERSION BEGIN;
    PUT SKIP FILE(SYSPRINT) EDIT ('**INVALID INPUT CARD**') (A);
    PUT SKIP FILE(SYSPRINT) EDIT (INSTR) (A(80));
    RCODE=-1;
    GOTO STOP;
   END;
```

```
ON ENDFILE (SYSIN) BEGIN;
  PUT SKIP FILE(SYSPRINT) EDIT
    (****ERROR - END OF FILE SYSIN ENCOUNTERED****) (A);
  RCODE=-1;
  GOTO STOP;
END;
ALLOCATE BLANK SET(P);
BLANK='';
DIRX.FILLER='';
I,J,K,RCODE,JJ,O=0;
CKEY.REC,LASTKEY=0;
RTYPE=-1;
RSIZE=70;
READ FILE(UNF) INTO(CONTROL) KEY(CKEY);
AFLAG='1'B;
DO WHILE(AFLAG);
  GET FILE(SYSIN) EDIT(INSTR) (COL(1),A(80));
  IF SUBSTR(INSTR,1,10)='****END OF COMMAND' THEN DO;
    AFLAG='0'B;
    RCODE=1;
  END;
  ELSE DO;
    GET STRING(INSTR) EDIT (XLEV,XCODE,XLAT,XLON,XNAME,XP,XB,XC,XL)
      (A(2),F(2,0),F(4,0),X(1),2 F(5,0),X(1),A(24),3 F(4,0),X(14),
      F(10,0));
/* CHECK FOR INVALID INPUT CARDS */
    IF XLEV < 1 | XLEV > CONTROL.NUMLEV THEN DO;
      AFLAG='0'B;
      RCODE=-2;
      PUT SKIP FILE(SYSPRINT) EDIT
        (****INVALID LEVEL NUMBER ON INPUT CARD****) (A);
      PUT SKIP FILE(SYSPRINT) EDIT (INSTR) (A(80));
    END;
    ELSE IF J=0 & XLEV>1 THEN DO;
      AFLAG='0'B; RCODE=-2;
      PUT SKIP FILE(SYSPRINT) EDIT
        (****FIRST INPUT CARD ISNOT A LEVEL-ONE DIRECTORY ENTRY****) (A);
      PUT SKIP FILE(SYSPRINT) EDIT (INSTR) (A(80));
    END;
    ELSE IF J>0 & XLEV=1 THEN DO;
      AFLAG='0'B; RCODE=-2;
      PUT SKIP FILE(SYSPRINT) EDIT
        (****ERROR - SECOND LEVEL-ONE DIRECTORY ENTRY READ****) (A);
      PUT SKIP FILE(SYSPRINT) EDIT (INSTR) (A(80));
    END;
    ELSE DO;
      IF XLEV = 1 THEN DO;
        J=1;
        BFLAG='1'B;
        /* FIND A CONTIGUOUS SET OF EMPTY BLOCKS FOR THE NEW      */
        /*   DIRECTORY ENTRIES                                    */
        /* THEN UPDATE CONTROL BLOCK INFO FOR THE FIRST OF THE    */
        /*   EMPTY BLOCKS                                         */
        DO I = 1 TO CONTROL.FILEBLKS WHILE(BFLAG);
          IF CONTROL.REC(I).RECTYPE = 0 THEN DO;
            CFLAG='1'B;
            DO K=I TO I+1 WHILE (CFLAG);
              IF CONTROL.REC(I+K).RECTYPE = 0 THEN CFLAG='0'B;
            END;
            IF CFLAG THEN DO;
              BFLAG='0'B;
              /* UPDATE CONTROL BLOCK INFO FOR A NEW LEVEL-ONE ENTRY */
              CONTROL.NUMONE,I = CONTROL.NUMONE + 1;
              CONTROL.ONE(I).CODE(1) = XCODE;
              CONTROL.ONE(I).NAME = XNAME;
              CONTROL.ONE(I).NUMBITS = 1;
              CONTROL.ONE(I).RECNO = L;
              CONTROL.ONE(I).DISPLACE = 1;
              CONTROL.REC(L).RECTYPE = RTYPE;
              CONTROL.REC(L).FREESPACE = CONTROL.REC(L).FREESPACE - RSIZE-1;
              CONTROL.REC(L).LOCATION = CONTROL.REC(L).LOCATION+RSIZE+1;
              K=L;
            END;
          END;
        END;
        IF BFLAG THEN DO;
          AFLAG='0'B; RCODE=-2;
          PUT SKIP FILE(SYSPRINT) EDIT
            (****FEWER THAN 11 BLOCKS REMAINING ON FILE****) (A);
        END;
      END;
      ELSE DO;
        J=J+1;
        I=REC;
```

```
            IF J=85 THEN DO;
              /* IF INPUT CARD IS THE 85TH TO BE READ IN, THEN REWRITE   */
              /*    FILE WITH 85 PREVIOUS AM DIRECTORY ENTRIES IN A BLOCK */
              REWRITE FILE(DAF) FROM(DIRX) KEY(REC);
              REC=REC+1;
              CONTROL.REC(L).RECTYPE=-1;
              BLANK=T;
              J=1;
            END;
            /* UPDATE CONTROL BLOCK INFO */
            CONTROL.REC(L).FREESPACE = CONTROL.REC(L).FREESPACE - RSIZE;
            CONTROL.REC(L).LOCATION = CONTROL.REC(L).LOCATION + RSIZE;
            CONTROL.ONE(1).NUMDIRS = CONTROL.ONE(1).NUMDIRS +1;
          END;
          /* DEFINE DIRECTORY INFO */
          DIRX.DIR(J).LEVNUM = ALEV;
          DIRX.DIR(J).CODENUM = XCODE;
          DIRX.DIR(J).LAT = FLAT;
          DIRX.DIR(J).LON = FLON;
          DIRX.DIR(J).DNAME = NAME;
          DIRX.DIR(J).LEVCODE = AL;
          DO K = 1 TO 6;
            DIRX.DIR(J).MODEL(K).CPOP = 0;
            DIRX.DIR(J).MODEL(K).APOP = -1;
            DIRX.DIR(J).MODEL(K).MDISP = 1;
          END;
          IF AP=-1 THEN DO;
            DIRX.DIR(J).PPOP = AP;
            DIRX.DIR(J).PDISP = 1;
          END;
          IF AP>0 THEN DO;
            D = ((AP-1)*RSIZE)+1;
/* 6384 = 78*84, WHICH ARE THE SIZE OF ONE DIRECTORY ENTRY AND THE   */
/*  MAXIMUM NUMBER OF ENTRIES ONE BLOCK CAN CONTAIN                  */
            II = FLOOR(D/6384);
            DIRX.DIR(J).PPOP = CONTROL.ONE(1).RECNUM + II;
            DIRX.DIR(J).PDISP = D - (II*6384);
          END;
          IF AR=-1 THEN DO;
            DIRX.DIR(J).RPOP = -1;
            DIRX.DIR(J).RDISP =1;
          END;
          IF AR>0 THEN DO;
            D = ((AR-1)*RSIZE)+1;
            II = FLOOR(D/6384);
            DIRX.DIR(J).RPOP = CONTROL.ONE(1).RECNUM + II;
            DIRX.DIR(J).RDISP = D - (II*6384);
          END;
          IF AC=-1 THEN DO;
            DIRX.DIR(J).CPOP = AC;
            DIRX.DIR(J).CDISP = 1;
          END;
          IF AC>0 THEN DO;
            D = ((AC-1)*RSIZE)+1;
            II = FLOOR(D/6384);
            DIRX.DIR(J).CPOP = CONTROL.ONE(1).RECNUM + II;
            DIRX.DIR(J).CDISP = D - (II*6384);
          END;
          DIRX.DIR(J).OPOP = -1;
          DIRX.DIR(J).ODISP = 1;
        END;
      END;
    END;
    IF REC>0 & BLANK=1 THEN DO;
      REWRITE FILE(DAF) FROM(DIRX) KEY(REC);
      REWRITE FILE(DAF) FROM(CONTROL) KEY(0);
    END;
    PUT SKIP FILE(SYSPRINT) EDIT (J,' DIRECTORY ENTRIES WRITTEN TO FILE')(A,F(5,0),A);
    FREE FLAPAK;
    STOP; RETURN;
    END YESJF04;
```

3.3.10 SUBPROGRAM STUBS

The subprogram stubs YESDF05, YESLS01, YESUD01, YESDE02 are dummy subroutines.

3.3.10.1 Linkages

N/A.

3.3.10.2 Interfaces

N/A.

3.3.10.3 Inputs

N/A.

3.3.10.4 Outputs

The message: DUMMY CALL TO YESXX00X.

3.3.10.5 Flow Chart

N/A.

3.3.10.6 Listing

Next page.

```
RUN NO. 15      DATE 11/12/76    TIME 0910          LISTING OF MODULE YESDF05
DESCRIPTION     DATA BASE PGM
MASTER FILE     W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER 10/13/76
LAST DATE COPIED   NONE
LAST UPDATE        NONE

PASSWORD        KLCL
PROGRAMMER      LEC
LANGUAGE        PLI
PROC PARAMETER  $NOJCL

   YESDF05:  PROC(SYSIN,SYSPRINT);
       DCL (SYSIN,SYSPRINT) FILE;
       PUT SKIP FILE(SYSPRINT) LIST('**DUMMY CALL YESDF05**');
       RETURN;
   END YESDF05;
```

```
RUN NO. 15      DATE 11/12/76    TIME 0910          LISTING OF MODULE YESLS01
DESCRIPTION     DATA BASE PGM
MASTER FILE     W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER 10/13/76
LAST DATE COPIED   NONE
LAST UPDATE        NONE

PASSWORD        VTCF
PROGRAMMER      LEC
LANGUAGE        PLI
PROC PARAMETER  $NOJCL

   YESLS01:  PROC(SYSIN,SYSPRINT);
       DCL (SYSIN,SYSPRINT) FILE;
       PUT SKIP FILE(SYSPRINT) LIST('**DUMMY CALL TO YESLS01**');
       RETURN;
   END YESLS01;
```

```
RUN NO. 15      DATE 11/12/76    TIME 0910          LISTING OF MODULE YESU001
DESCRIPTION     DATA BASE PGM
MASTER FILE     W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER 10/13/76
LAST DATE COPIED   NONE
LAST UPDATE        NONE

PASSWORD        BLBV
PROGRAMMER      LEC
LANGUAGE        PLI
PROC PARAMETER  $NOJCL

   YESU001:  PROC(SYSIN,SYSPRINT);
       DCL (SYSIN,SYSPRINT) FILE;
       PUT SKIP FILE(SYSPRINT) LIST('**DUMMY CALL TO YESU001**');
       RETURN;
   END YESU001;
```

RUN NO. IS DATE 11/12/76 TIME 0910 LISTING OF MODULE YESDE01

DESCRIPTION DATA BASE PGM

MASTER FILE W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER 10/13/76
LAST DATE COPIED NONE
LAST UPDATE NONE

PASSWORD HLRX
PROGRAMMER LEC
LANGUAGE PLI
PROC PARAMETER SNOJCL

```
YESDE01: PROC(SYSIN,SYSPRINT);
         DCL (SYSIN,SYSPRINT) FILE;
         PUT SKIP FILE(SYSPRINT) LIST('**DUMMY CALL TO YESDE01**');
         RETURN;
     END YESDE01;
```

3.3.11 UPDATING THE DATA BASE (UPDDATA)

UPDDATA will enter or modify data in the data base.

3.3.11.1 Linkages

None.

3.3.11.2 Interfaces

A directory must have been defined for the area to be updated.

3.3.11.3 Inputs

Update data cards.

3.3.11.4 Outputs

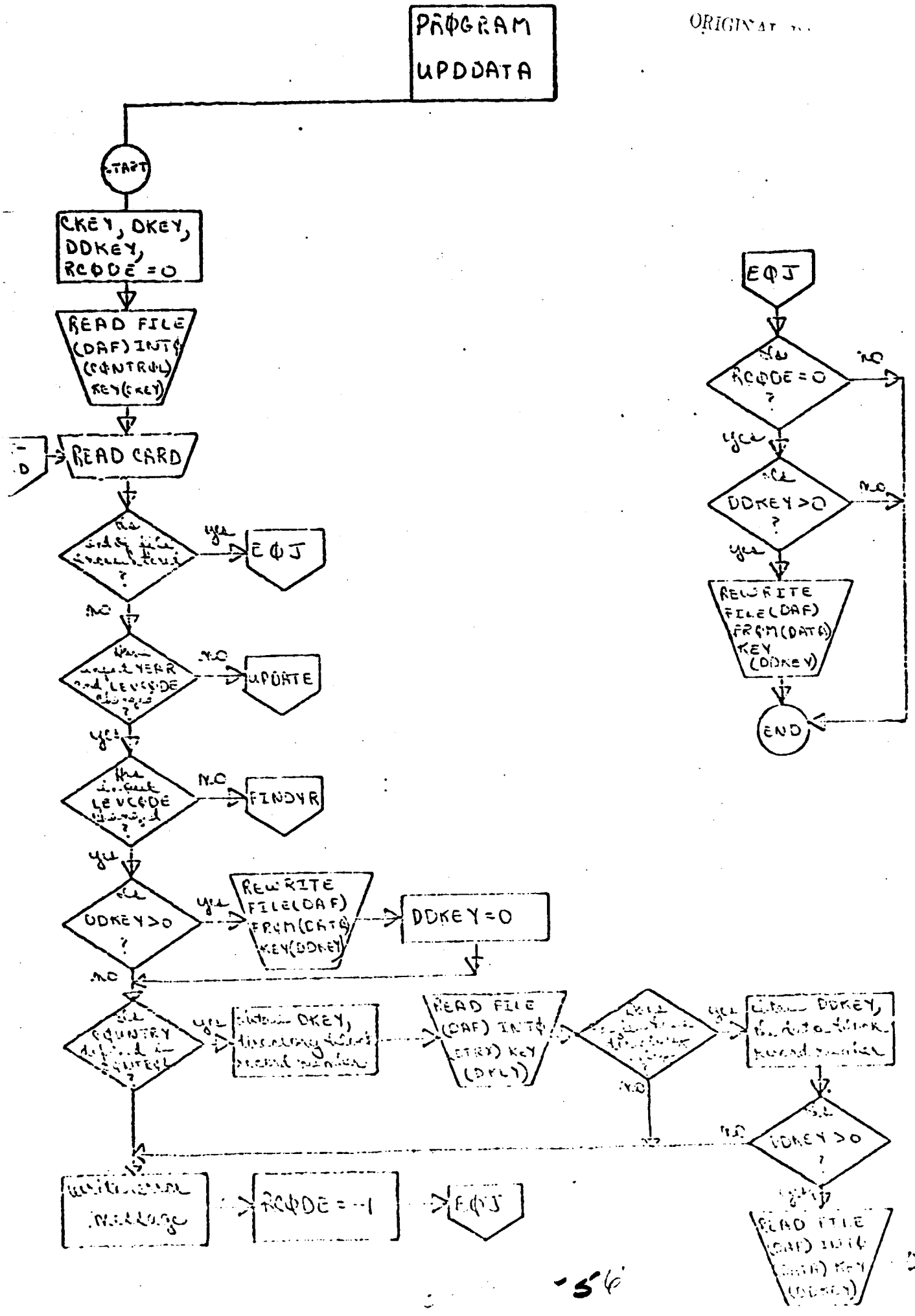
Updated data base.

3.3.11.5 Flow Chart

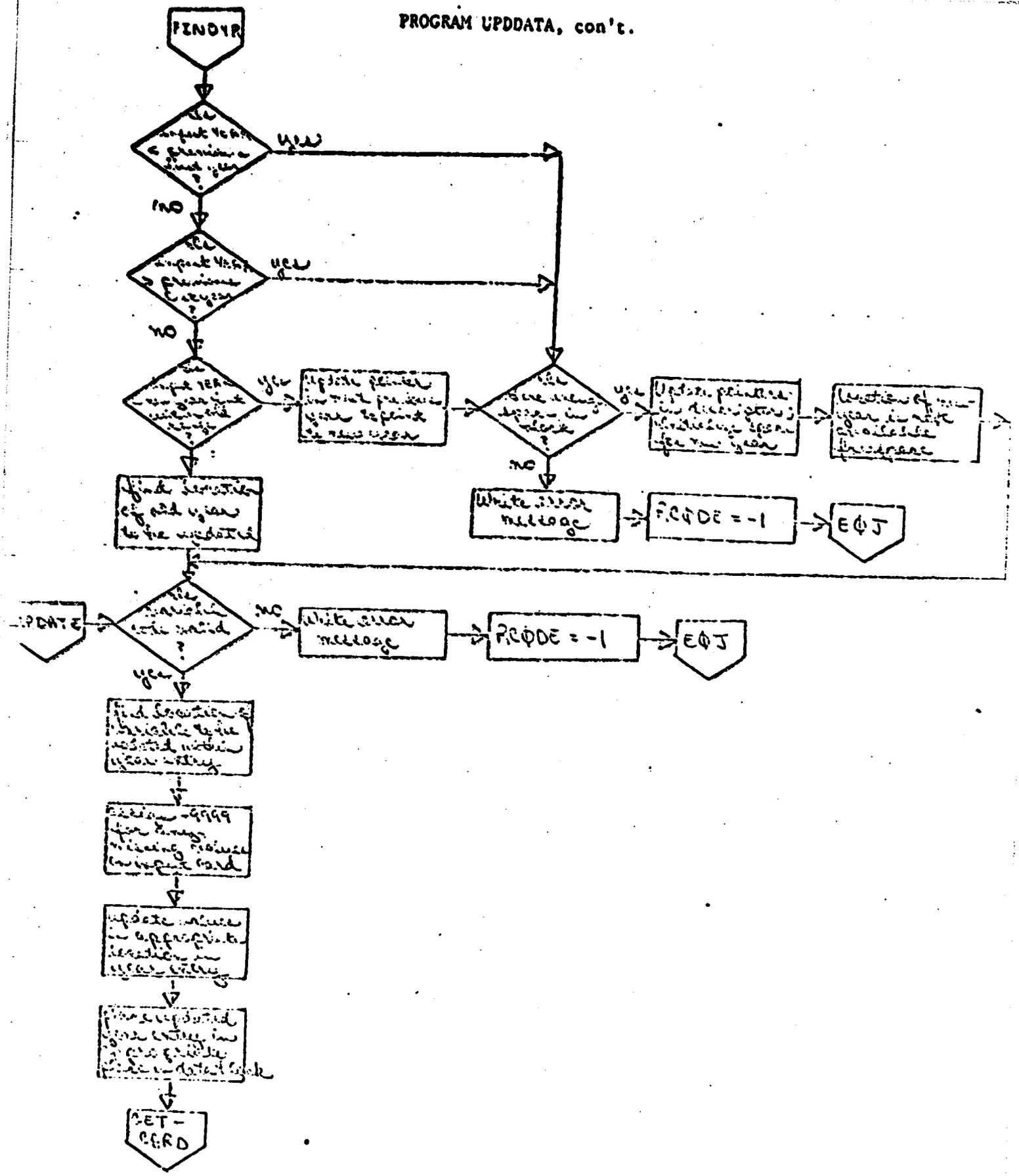
Next page.

3.3.11.6 Listing

Follows flow chart.


PROGRAM
UPDDATA
START
CKEY, OKEY,
DDKEY,
RCODE = 0
READ FILE
(DAF) INTO
(CONTROL)
KEY(CKEY)
READ CARD
EOJ
UPDATE
FINDYR
REWRITE
FILE(DAF)
FROM(DATA)
KEY
(DDKEY)
DDKEY = 0
READ FILE
(DAF) INTO
(ENTRY) KEY
(DKEY)
RCODE = -1
END

PROGRAM UPDDATA, con't.



```
PL/I OPTIMIZING COMPILER                UPDDATA: PROC OPTIONS(MAIN);

   STMT LEV NT

                        /*  UNPRINTABLE CHARACTER AND APPEARS AS A BLANK                */
                        /*                                                              */
     14    1  0    DCL ZERO CHAR(128) VAR INIT((128)' ');
     15    1  0    DCL YRDATA CHAR(124) VAR;
     16    1  0    DCL METDATA(12) FIXED BIN(15,0) BASED(P1);
     17    1  0    DCL MET CHAR(24) BASED(P1);
     18    1  0    DCL YLDDATA(4) FIXED BIN(31,0) BASED(P2);
     19    1  0    DCL YLD CHAR(16) BASED(P2);
     20    1  0    DCL PNTRDATA(4) FIXED BIN(15,0) BASEDIP3);
     21    1  0    DCL PNTR CHAR(8) BASED(P3);
     22    1  0    DCL AFLAG BIT(1);
     23    1  0    DCL (CKEY,DKEY,DDKEY,DNUM,DDISP,I,J,FSTYR,LSTYR,YRSLEFT,DIFF,DSP,
                     SPACE,LENGTH,ELEMENTS,BEFORE,     NEWYR,RCODE,NXTDISP,FORMERYR,
                     FORMERDSP)
                    FIXED BIN(15,0);
     24    1  0    DCL LEVS FIXED BIN(31,0);
     25    1  0    DCL (P1,P2,P3,Q,R) POINTER;
     26    1  0    DCL 01 POINTER;
     27    1  0    ON ENDFILE(CARDS) GOTO EOJ;
     28    1  0    CKEY,DKEY,DDKEY,DDISP,DNUM,LEVS,NEWYR,RCODE,FSTYR,LSTYR,YRSLEFT=0;
     29    1  0    DIFF,DSP,LENGTH,ELEMENTS,BEFORE=0;
     30    1  0    READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
     31    1  0    ALLOCATE INPUT;
     32    1  0    ALLOCATE INSTR;
     33    1  0    GETCARD: GET FILE(CARDS) EDIT(INSTR) (COL(1),A(80));
     34    1  0      GET STRING(INSTR) EDIT (COUNTRY) (X(70),F(2,0));
     35    1  0      GET STRING(INSTR) EDIT(INPUT.INYR,INPUT.INCD,DEF.INLVLCD)
                       (X(3),F(4,0),F(3,0),X(60),F(10,0));
     36    1  0      IF INPUT.INCD<100 THEN GET STRING(INSTR) EDIT
                      ((INPUT.INMON(I) DO I=1 TO 12)) (X(10),12 F(5,0));
     37    1  0      IF INPUT.INCD>100 THEN GET STRING(INSTR) EDIT
                      ((DEF.INAGDTA(I) DO I=1 TO 6)) (X(10),6 F(10,0));
                        /*                                                            */
                        /* CHECK IF INPUT ENTRY AND YEAR ARE THE SAME AS THOSE ON     */
                        /*  THE PREVIOUS CARD. IF BOTH ARE EQUAL THEN GO TO SECTION   */
                        /*  WHICH CHECKS FOR NEW VARIABLE CODE.  IF ONLY YEAR IS      */
                        /*  CHANGED THEN GO TO SECTION WHICH FINDS NEW YEAR BLOCK.    */
                        /*  IF BOTH ARE CHANGED AND THIS ISNOT THE FIRST INPUT CARD   */
                        /*  THEN REWRITE UPDATED DATA BLOCK AND START PROCESS OVER    */
                        /*                                                            */
     38    1  0      IF DEF.INLVLCD=LEVS & INPUT.INYR=NEWYR THEN GOTO UPDATE;
     39    1  0      IF DEF.INLVLCD=LEVS THEN GOTO FINDYR;
     40    1  0      IF DDKEY>0 THEN DO;
     41    1  1        REWRITE FILE(DAF) FROM(DATA) KEY(DDKEY);
     42    1  1        PUT SKIP(2) FILE(SYSPRINT) EDIT ('*** FILE UPDATED FOR REGION',
                         LEVS) (A,F(11,0));
     43    1  1        DKEY,DDKEY=0;
     44    1  1        FREE DUMMY;
     45    1  1      END;
     46    1  0      LEVS=DEF.INLVLCD;  NEWYR=INPUT.INYR;
     48    1  0      ALLOCATE DUMMY;
     49    1  0      AFLAG='1'B;
                        /*                                                            */
                        /* FIND DIRECTORY BLOCK FOR COUNTRY DEFINED ON INPUT CARD     */
                        /*                                                            */
     50    1  0      DO I=1 TO CONTROL.NUMONE WHILE(AFLAG);
     51    1  1        IF COUNTRY=CONTROL.ONE(I).CODE1NUM THEN DO;
     52    1  2          AFLAG='0'B;
     53    1  2          DKEY=CONTROL.ONE(I).RECNUM;
     54    1  2          DNUM=CONTROL.ONE(I).NUMDIRS;
     55    1  2        END;
     56    1  1      END;
     57    1  0      IF DKEY=0 THEN DO;
     58    1  1        PUT SKIP(2) FILE(SYSPRINT) EDIT ('***NO DIRECTORY BLOCK FOUND FOR',
                         ' COUNTRY CODE ',COUNTRY      ,' ***') (A,A,F(3,0),A);
     59    1  1        RCODE=-1;
     60    1  1        GOTO EOJ;
     61    1  1      END;
     62    1  0      READ FILE(DAF) INTO(DIRX) KEY(DKEY);
     63    1  0      J=0;
     64    1  0      AFLAG='1'B;
                        /*                                                            */
                        /* FIND DATA DESCRIPTOR AND DATA BLOCK FOR INPUT ENTRY        */
                        /*                                                            */
     65    1  0      DO I=1 TO DNUM WHILE(AFLAG);
     66    1  1        J=J+1;
     67    1  1        IF J=45 THEN DO;
     68    1  2          DKEY=DKEY+1;
     69    1  2          READ FILE(DAF) INTO(DIRX) KEY(DKEY);
     70    1  2          J=1;
     71    1  2        END;
```

```
   72   1  1        IF DEF.INLVLCD=DIR(J).LEVCODE THEN DO;
   73   1  2          AFLAG='0'B;
   74   1  2          DDKEY=DIR(J).DREC;
   75   1  2          DDDISP=DIR(J).DDISP;
   76   1  2        END;
   77   1  1      END;
   78   1  0      IF DDKEY=0 THEN DO;
   79   1  1        PUT SKIP(2) FILE(SYSPRINT) EDIT ('***NO DATA BLOCK FOUND FOR',
                    ' INPUT CODE ',DEF.INLVLCD,' ***') (A,A,F(11,0),A);
   80   1  1        RCODE=-1;
   81   1  1        GOTO EOJ;
   82   1  1      END;
   83   1  0      READ FILE(DAF) INTO(DATA) KEY(DDKEY);
   84   1  0      DUMMY=SUBSTR(DATA,DDDISP,336);
   85   1  0  FINDYR:  ALLOCATE PNTR;
                    /*                                                    */
                    /* FIND FIRST AND LAST CHRONOLOGICAL YEARS DEFINED AND ALSO*/
                    /*  DISPLACEMENT OF FREESPACE WITHIN DATA BLOCK        */
                    /*                                                    */
   86   1  0  IF DESC.NUMBYRS=0 THEN SUBSTR(DATA,DESC.FSTDISP,8)=ZERO;
   87   1  0    PNTP=SUBSTR(DATA,DESC.FSTDISP,8);
   88   1  0    FSTYR=PNTRDATA(1);
   89   1  0    PNTR=SUBSTR(DATA,DESC.LSTDISP,8);
   90   1  0    LSTYR=PNTRDATA(1);
   91   1  0    DSP=(DESC.NUMBYRS*DESC.BLKSIZE)+DDDISP+336;
                    /*                                                    */
                    /* IF INPUT YEAR IS LESS THAN FIRST YEAR, THEN CHECK FOR  */
                    /*  SPACE IN DATA BLOCK, CHANGE POINTER IN DESCRIPTOR, AND */
                    /*  INITIALIZE POINTERS AND DATA FOR NEW FIRST YEAR    */
                    /*                                                    */
   92   1  0      IF INPUT.INYR<FSTYR THEN DO;
   93   1  1        IF DESC.NUMBYRS=DESC.TOTBLKS THEN DO;
   94   1  2          PUT SKIP(2) FILE(SYSPRINT) EDIT('*** NOT ENOUGH YEAR BLOCKS',
                        ' ALLOCATED TO ACCOMMODATE AN EXTRA YEAR ***') (A,A);
   95   1  2          RCODE=-1;
   96   1  2          GOTO EOJ;
   97   1  2        END;
   98   1  1        PNTRDATA(1)=INPUT.INYR;
   99   1  1        PNTRDATA(2)=DDKEY;
  100   1  1        PNTRDATA(3)=DESC.FSTDISP;
  101   1  1        PNTRDATA(4)=0;
  102   1  1        SUBSTR(DATA,DSP,8)=PNTR;
  103   1  1        SUBSTR(DATA,DSP+8,DESC.BLKSIZE-8)=ZERO;
  104   1  1        DESC.NUMBYRS=DESC.NUMBYRS+1;
  105   1  1        DESC.FSTDISP=DSP;
  106   1  1      END;
                    /*                                                    */
                    /* IF INPUT YEAR IS GREATER THAN LAST YEAR, THEN CHECK FOR */
                    /*  SPACE IN DATA BLOCK, CHANGE POINTER IN DESCRIPTOR, AND */
                    /*  INITIALIZE POINTERS AND DATA FOR NEW LAST YEAR     */
                    /*                                                    */
  107   1  0      ELSE IF INPUT.INYR>LSTYR THEN DO;
  108   1  1        IF DESC.NUMBYRS=DESC.TOTBLKS THEN DO;
  109   1  2          PUT SKIP(2) FILE(SYSPRINT) EDIT('*** NOT ENOUGH YEAR BLOCKS',
                        ' ALLOCATED TO ACCOMMODATE AN EXTRA YEAR ***') (A,A);
  110   1  2          RCODE=-1;
  111   1  2          GOTO EOJ;
  112   1  2        END;
  113   1  1        PNTRDATA(1)=INPUT.INYR;
  114   1  1        PNTRDATA(2)=-1;
  115   1  1        PNTRDATA(3)=0;
  116   1  1        PNTRDATA(4)=0;
  117   1  1        SUBSTR(DATA,DSP,8)=PNTR;
  118   1  1        SUBSTR(DATA,DSP+8,DESC.BLKSIZE-8)=ZERO;
  119   1  1        PNTRDATA(1)=LSTYR;
  120   1  1        PNTRDATA(2)=DDKEY;
  121   1  1        PNTRDATA(3)=DSP;
  122   1  1        PNTRDATA(4)=0;
  123   1  1        IF DESC.NUMBYRS>0 THEN SUBSTR(DATA,DESC.LSTDISP,8)=PNTR;
  124   1  1        DESC.NUMBYRS=DESC.NUMBYRS+1;
  125   1  1        DESC.LSTDISP=DSP;
  126   1  1      END;
                    /*                                                    */
                    /* IF INPUT YEAR EQUALS A PREVIOUSLY DEFINED YEAR, FIND   */
                    /*  ITS DISPLACEMENT WITHIN DATA BLOCK.  IF NOT, CHANGE   */
                    /*  NEXT YEAR POINTER IN APPROPRIATE YEAR BLOCK AND    */
                    /*  INITIALIZE POINTERS AND DATA FOR NEW YEAR          */
                    /*                                                    */
  127   1  0      ELSE DO;
  128   1  1        AFLAG='1'B;
  129   1  1        NXTDISP=DESC.FSTDISP;
  130   1  1        DO I=1 TO DESC.NUMBYRS WHILE(AFLAG);
  131   1  2          PNTR=SUBSTR(DATA,NXTDISP,8);
  132   1  2          IF INPUT.INYR=PNTRDATA(1) THEN DO;
  133   1  3            DSP=NXTDISP;
  134   1  3            AFLAG='0'B;
  135   1  3          END;
```

```
136  1  2          ELSE IF INPUT.INYR<PNTRDATA(1) THEN DO;
137  1  2            PNTRDATA(1)=FORMERYR;
138  1  2            PNTRDATA(2)=DDKEY;
139  1  2            PNTRDATA(3)=DSP;
140  1  2            PNTRDATA(4)=0;
141  1  2            SUBSTR(DATA,FORMERDSP,8)=PNTR;
142  1  2            SUBSTR(DATA,DSP+8,DESC.BLKSIZE-8)=ZERO;
143  1  2            PNTRDATA(1)=INPUT.INYR;
144  1  2            PNTRDATA(2)=DDKEY;
145  1  2            PNTRDATA(3)=NXTDISP;
146  1  2            PNTRDATA(4)=0;
147  1  2            SUBSTR(DATA,DSP,8)=PNTR;
148  1  2            DESC.NUMBYRS=DESC.NUMBYRS+1;
149  1  2            AFLAG='0'B;
150  1  2          END;
151  1  2          ELSE DO;
152  1  2            FORMERYR=PNTRDATA(1);
153  1  2            FORMERDSP=NXTDISP;
154  1  2            NXTDISP=PNTRDATA(3);
155  1  2          END;
156  1  2        END;
157  1  1      END;
158  1  0      SUBSTR(DATA,DDDISP,336)=DUMMY;
159  1  0      YRDATA=ZERO;
160  1  0      YRDATA=SUBSTR(DATA,DSP,DESC.BLKSIZE);
161  1  0    UPDATE: SPACE=1;  LENGTH=0;
163  1  0      AFLAG='1'B;
                /*                                                          */
                /* FIND LOCATION AND LENGTH OF FIELD AND NUMBER OF          */
                /*  SUBELEMENTS FOR VARIABLE CODE DEFINED ON INPUT CARD     */
                /*                                                          */
164  1  0      DO I=1 TO DESC.NUMBCODE WHILE(AFLAG);
165  1  1        IF INPUT.INCD=DESC.DCODE(I).CODENUMB THEN DO;
166  1  2          LENGTH=DESC.DCODE(I).NUMSELEM*DESC.DCODE(I).ELEMSIZE;
167  1  2          ELEMENTS=DESC.DCODE(I).NUMSELEM;
168  1  2          BEFORE=SPACE;
169  1  2          AFLAG='0'B;
170  1  2        END;
171  1  1        SPACE=SPACE+(DESC.DCODE(I).NUMSELEM*DESC.DCODE(I).ELEMSIZE);
172  1  1      END;
173  1  0      IF AFLAG THEN DO;
174  1  1        PUT SKIP(2) FILE(SYSPRINT) EDIT ('***INVALID VARIABLE CODE',
                  INPUT.INCD,' ENCOUNTERED***') (A,F(5,0),A);
175  1  1        RCODE=-1;
176  1  1        GOTO EOJ;
177  1  1      END;
178  1  0      IF INPUT.INCD<100 THEN DO;
                /*                                                          */
                /* UPDATE INFORMATION IF VARIABLE IS METEOROLOGICAL         */
                /*                                                          */
179  1  1        ALLOCATE MET;
180  1  1        DO I=1 TO ELEMENTS;
181  1  2          IF DEFA.ACHAR(I)='    ' THEN INPUT.INMON(I)=-9999;
182  1  2          METDATA(I)=INPUT.INMON(I);
183  1  2        END;
184  1  1        SUBSTR(YRDATA,BEFORE,LENGTH)=SUBSTR(MET,1,LENGTH);
185  1  1        FREE MET;
186  1  1      END;
187  1  0      IF INPUT.INCD>100 THEN DO;
                /*                                                          */
                /* UPDATE INFORMATION IF VARIABLE IS YIELD-TYPE             */
                /*                                                          */
188  1  1        ALLOCATE YLD;
189  1  1        DO I=1 TO ELEMENTS;
190  1  2          IF DEFX.ACHARX(I)='          ' THEN DEF.INAGDTA(I)=-9999;
191  1  2          YLDDATA(I)=DEF.INAGDTA(I);
192  1  2        END;
193  1  1        SUBSTR(YRDATA,BEFORE,LENGTH)=SUBSTR(YLD,1,LENGTH);
194  1  1        FREE YLD;
195  1  1      END;
                /*                                                          */
                /* PLACE CHANGES BACK IN DATA BLOCK AND LOOP FOR NEW CARD   */
                /*                                                          */
196  1  0      SUBSTR(DATA,DSP,DESC.BLKSIZE)=YRDATA;
197  1  0    GOTO GETCARD;
198  1  0    EOJ: IF RCODE=0 THEN DO;
199  1  1           IF DDKEY>0 THEN REWRITE FILE(DAF) FROM(DATA) KEY(DDKEY);
200  1  1           PUT SKIP(2) FILE(SYSPRINT) EDIT ('**END OF DATA UPDATE**') (A);
201  1  1         END;
202  1  0    FREE INSTR;
203  1  0    FREE INPUT;
204  1  0    FREE DUMMY;
205  1  0    END UPDDATA;
```

3.3.12 INITIAL DATA LOADERS

The four programs USA, USSR, CANADA, and AUSARG are provided, one each for the USA, USSR, and Canada and a common loader for Australia/Argentina, to initially load the data base.

3.3.12.1 Linkages

All loaders call GETDIR.

3.3.12.2 Interfaces

Directory entries must exist for all data entered.

3.3.12.3 Inputs

Data cards.

3.3.12.4 Outputs

Initial data load of data base.

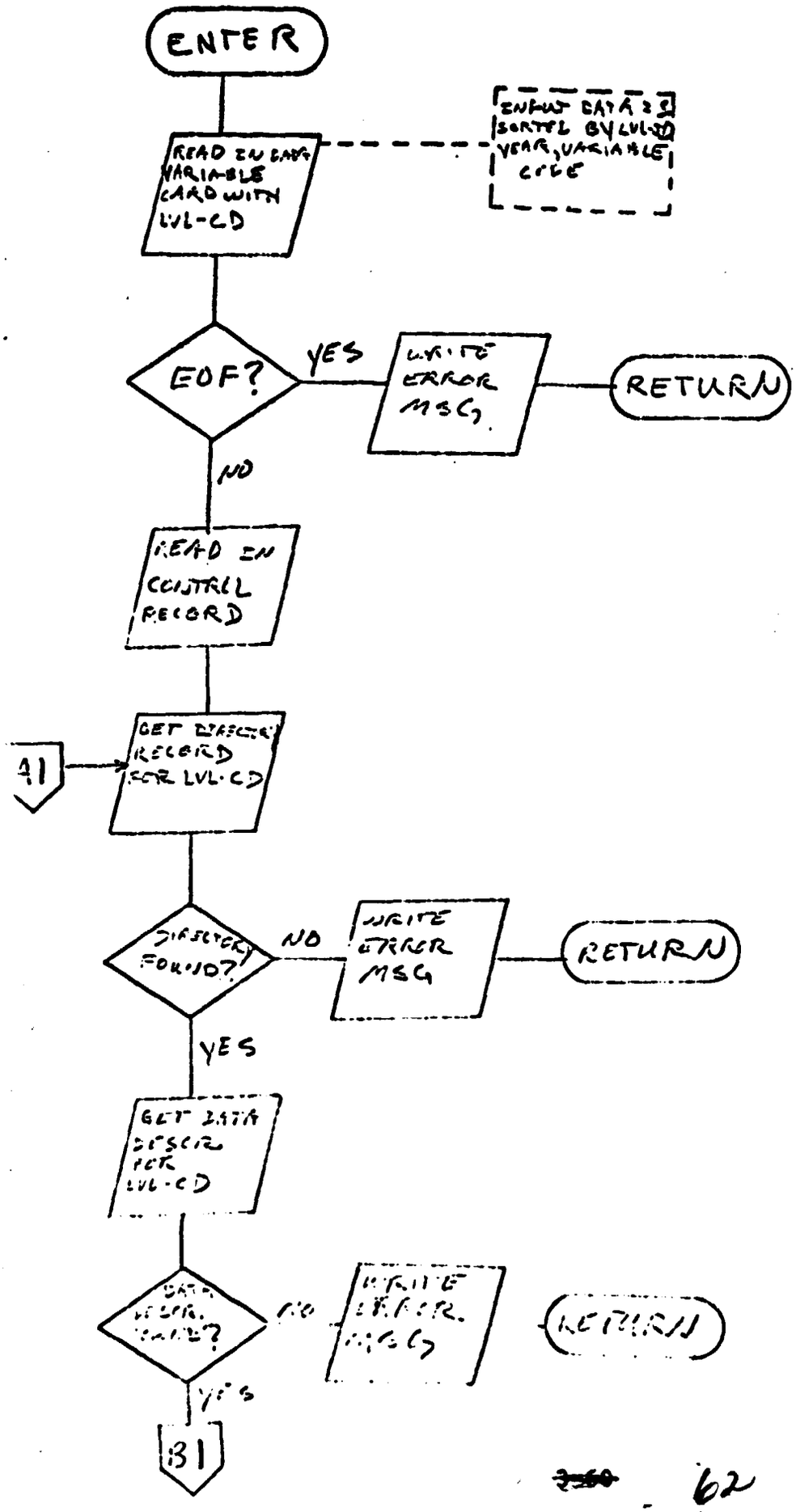
3.3.12.5 Flow Chart

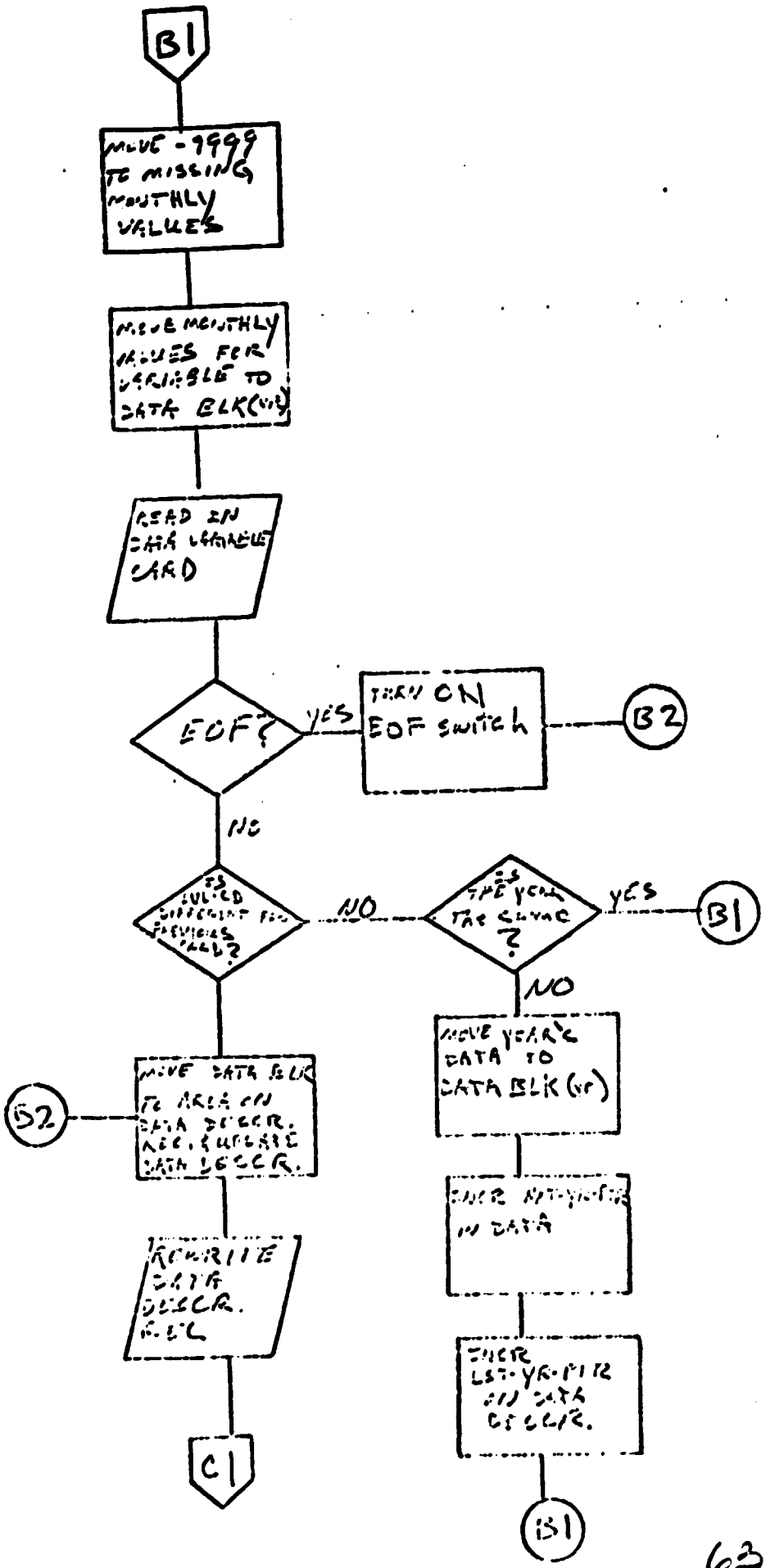
A common flow chart is on the next page.

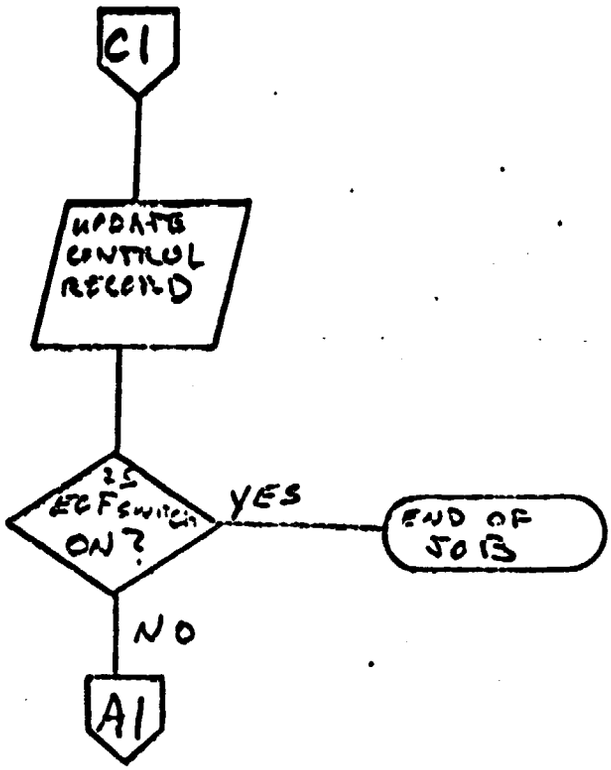
3.3.12.6 Listing

Follows flow chart.

LOAD MET DATA




B1
MOVE -9999 TO MISSING MONTHLY VALUES
MOVE MONTHLY VALUES FOR VARIABLE TO DATA BLK(VR)
READ IN DATA VARIABLE CARD
EOF?
YES
TURN ON EOF switch
B2
NO
IS VALUE DIFFERENT FROM PREVIOUS VALUE?
NO
IS THIS YEAR THE SAME?
YES
B1
NO
MOVE YEAR'S DATA TO DATA BLK (YR)
ENTER NEW YEAR IN DATA
ENTER LST-YR-PTR AND DATA DESCR.
B1
B2
MOVE DATA BLK TO AREA IN DATA DESCR. REC. FOR LATEST DATA DESCR.
REWRITE DATA DESCR. REC.
C1


C1
UPDATE CONTROL RECORD
SS ECF switch ON?
YES
END OF JOB
NO
A1

RUN NO. 19 DATE 11/12/75 TIME 0920 LISTING OF MODULE [illegible]

DESCRIPTION DATA BASE PGM

```
MASTER FILE          ..EDS.CCEA.LEC.LIBR
[illegible]          [illegible]
LAST DATE COPIED     NONE
LAST UPDATE          [illegible]

PASSWORD             [illegible]
PROGRAMMER           LEC
LANGUAGE             [illegible]
PROC PARAMETER       ENDJCL

   [illegible] :
      DCL 1 CONTROL,
            2 [illegible],
            2 NUMPASS FIXED BIN(15,0),
            2 [illegible],
            2 NUMLEV FIXED BIN(15,0),
            2 [illegible],
            2 NUMCODE FIXED BIN(15,0),
            2 [illegible],
               3 COURSNUM FIXED BIN(15,0),
               3 [illegible],
               3 PASS FIXED BIN(15,0),
               3 [illegible],
               3 [illegible] CHAR(2),
               3 [illegible],
            2 NUMONE FIXED BIN(15,0),
            2 [illegible],
               3 [illegible] FIXED BIN(15,0),
               3 [illegible],
               3 RECNUM FIXED BIN(15,0),
               3 [illegible],
               3 NAME CHAR(24),
            2 [illegible],
            2 REC(50),
               3 [illegible],
               3 PASSPACE FIXED BIN(15,0),
               3 [illegible];
      DCL 1 DISK,
            2 [illegible],
               3 LEVNUM FIXED BIN(15,0),
               3 [illegible],
               3 LAT FIXED BIN(15,0),
               3 [illegible],
               3 [illegible] CHAR(24),
               3 [illegible],
               3 [illegible] FIXED BIN(15,0),
               3 [illegible],
               3 [illegible] FIXED BIN(15,0),
               3 [illegible],
               3 [illegible] FIXED BIN(15,0),
               3 [illegible],
               3 COTSP FIXED BIN(15,0),
               3 [illegible],
               3 [illegible](4),
                  4 [illegible],
                  4 [illegible] FIXED BIN(15,0),
                  4 [illegible];
         2 FILLER CHAR([illegible]);
      DCL 1 [illegible],
            2 [illegible]         PIC '(7)X',
            2 [illegible]         PIC '[illegible]',
            2 [illegible](12)     PIC 'SSSSV',
            2 [illegible],
               3 [illegible]      PIC '[illegible]',
               3 [illegible]      PIC '[illegible]',
               3 [illegible]      PIC '[illegible]',
               3 [illegible]      PIC '[illegible]',
               3 [illegible]STRA  PIC '[illegible]';
      DCL 1 [illegible],
            2 FILE                CHAR(10),
            2 [illegible]         PIC '[illegible]',
            2 [illegible]         PIC '(10)[illegible]';
      DCL 1 [illegible],
            2 FILE                CHAR(10),
            2 [illegible]         CHAR([illegible]),
            2 FILE                CHAR(10);
      DCL 1 [illegible],
            2 FILE                CHAR(10),
            2 [illegible]         CHAR([illegible]),
            2 FILE                CHAR(10);
```

```
      DCL 1 RUSSIA_DATA BASED(P),
            2 RUS_YR          FIXED BIN(15),
            2 RUS_NXT_YR      FIXED BIN(15),
            2 RUS_NXT_DISP    FIXED BIN(15),
            2 RESERV2         FIXED BIN(15),
            2 RUS_TEMP(12)    FIXED BIN(15),
            2 RUS_PRECIP(12)  FIXED BIN(15),
            2 RUS_HARV(4)     FIXED BIN(31),
            2 RUS_PROD(4)     FIXED BIN(31);
      DCL DATA_OUT CHAR(88) BASED(P);
      ALLOCATE RUSSIA_DATA;
      RUSSIA_DATA.RESERV2 = 0;
      ALLOCATE INPUT_DATA;
      DCL 1 DATA_BLK,
            2 DATADESC,
              3 ID                FIXED BIN(31,0),
              3 WMO               FIXED BIN(31,0),
              3 LATI              FIXED BIN(15,0),
              3 LONGI             FIXED BIN(15,0),
              3 ELEV              FIXED BIN(15,0),
              3 TOTALBLKS_ALLOC   FIXED BIN(15,0),
              3 NUMBYRS_USED      FIXED BIN(15,0),
              3 BLOCKSIZE         FIXED BIN(15,0),
              3 FSTRECNO          FIXED BIN(15,0),
              3 FSTDISP           FIXED BIN(15,0),
              3 LSTRECNO          FIXED BIN(15,0),
              3 LSTDISP           FIXED BIN(15,0),
              3 RESERVED          CHAR(18),
              3 NUMSCODE          FIXED BIN(15,0),
            3 DCODE(12),
              4 CODENUM           FIXED BIN(15,0),
              4 NUMSELEM          FIXED BIN(15,0),
              4 ELEMSIZE          FIXED BIN(15,0),
              4 NUMSCODE          FIXED BIN(15,0),
              4 SUBCODE(8)        FIXED BIN(15,0),
          2 YR_BLK(32)  CHAR(88),
          2 FIL         CHAR(38) INIT(' '),
          2 DATADES2 LIKE DATADESC,
          2 YR_BLK2(32) CHAR(88),
          2 FIL2 CHAR(68) INIT(' ');
      DCL HOLD_LVLCD PIC'(10)9';
      DCL HOLD_YR PIC '(7)9';
      DCL DIR_CD FIXED BIN(31);
      DCL FILDATA CHAR(88) BASED(P1);
      DCL 1 DUMRUS BASED(P1),
            2 FILYR FIXED BIN(15),
            2 FIL_NXT_YR FIXED BIN(15),
            2 FIL_NXT_DISP FIXED BIN(15),
            2 FIL_RES2 FIXED BIN(15),
            2 FIL_TEMP(12)  FIXED BIN(15),
            2 FIL_PRECIP(12) FIXED BIN(15),
            2 FIL_HARV(4) FIXED BIN(31),
            2 FIL_PROD(4) FIXED BIN(31);
      ALLOCATE FILDATA;
      FIL_RESV = 0;
      FIL_TEMP = -9999;
      FIL_PRECIP = -9999;
      FIL_HARV = 0;
      FIL_PROD = 0;
      DCL (OKEY,CKEY,ODKEY,DIR#,RC,YRCTR) FIXED BIN(15);
      CKEY=0;
      DCL STRTYR FIXED BIN(15);
      STRTYR = 1956;
      YRCTR=0;
      RESERV2 = 0;
      RUS_TEMP = -9999;
      RUS_PRECIP = -9999;
      RUS_HARV = 0;
      RUS_PROD = 0;
      DCL EOF_SW CHAR(1) INIT('0');
      DCL GETDIR EXTERNAL ENTRY;
      DCL CARDS FILE RECORD INPUT;
      DCL DAF FILE RECORD DIRECT UPDATE KEYED ENV(REGIONAL(1));
      ON CONVERSION GO TO TERM_ERR;
      ON ENDFILE(CARDS) BEGIN;
        EOF_SW = '1';
        GO TO NEW_STRATA;
        END;
  FRST_RD:  ;
      READ FILE(CARDS) INTO(INPUT_DATA);
      INSTRA = '00';
     INYR = TRANSLATE(INYR,'0',' ');
     INCD = TRANSLATE(INCD,'0',' ');
     INLVLCD = TRANSLATE(INLVLCD,'0',' ');
      HOLD_YR = INYR;
```

```
      HOLD_LVLCD = INLVLCD;
      DIR_CD = HOLD_LVLCD;
      READ FILE(DAFT) INTO(CONTROL) KEY(CKEY);
      CALL GETDIR(DIR_CD,DIRX,CONTROL,DAF,DKEY,DIR#,RC);
      DDKEY = DREC(DIR#);
      READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
      GO TO CHK_CDS;
GET_DATA:  ;
      READ FILE(CARDS) INTO(INPUT_DATA);
      INSTRA = '00';
      INYR = TRANSLATE(INYR,'0',' ');
      INCD = TRANSLATE(INCD,'0',' ');
      INLVLCD = TRANSLATE(INLVLCD,'0',' ');
      IF INLVLCD ¬= HOLD_LVLCD THEN GO TO NEW_STRATA;
      IF INYR < HOLD_YR THEN GO TO SEQ_ERR;
      IF INYR > HOLD_YR THEN GO TO PUT_DATA;
CHK_CDS:  ;
      IF INCD = 5 THEN GO TO MV_RUS_PRECIP;
      IF INCD = 35 THEN GO TO MV_RUS_T;
      IF INCD = 101 THEN GO TO MV_RUS_HARV;
      IF INCD = 103 THEN GO TO MV_RUS_PROD;
      GO TO ERRCD;
MV_RUS_PRECIP:  ;
      DO N = 1 TO 12;
        IF ACHAR(N) = '    ' THEN INMON(N) = -9999;
        RUS_PRECIP(N) = INMON(N);
        END;
      GO TO GET_DATA;
MV_RUS_T:  ;
      DO N = 1 TO 12;
        IF ACHAR(N) = '    ' THEN INMON(N) = -9999;
        RUS_TEMP(N) = INMON(N);
        END;
      GO TO GET_DATA;
MV_RUS_HARV:  ;
      IF ACHARX(1) = '          ' THEN INAGDTA(1) = -9999;
      IF ACHARX(2) = '          ' THEN INAGDTA(2) = -9999;
      IF INREG = 01 THEN DO;
        RUS_HARV(1) = INAGDTA(1);
        END;
      IF INREG = 02 THEN DO;
        RUS_HARV(1) = INAGDTA(1);
        RUS_HARV(2) = INAGDTA(2);
        END;
      IF INREG = 03 THEN DO;
        RUS_HARV(1) = INAGDTA(1);
        END;
      GO TO GET_DATA;
MV_RUS_PROD:  ;
      IF ACHARX(1) = '          ' THEN INAGDTA(1) = -9999;
      IF ACHARX(2) = '          ' THEN INAGDTA(2) = -9999;
      IF INREG = 01 THEN DO;
        RUS_PROD(1) = INAGDTA(1);
        END;
      IF INREG = 02 THEN DO;
        RUS_PROD(1) = INAGDTA(1);
        RUS_PROD(2) = INAGDTA(2);
        END;
      IF INREG = 03 THEN DO;
        RUS_PROD(1) = INAGDTA(1);
        END;
      GO TO GET_DATA;
PUT_DATA:  ;
      IF YRCTR > 22 THEN GO TO NO_MV_RUS;
      YRCTR = YRCTR + 1;
      STRTYR = STRTYR + 1;
      RUS_YR = HOLD_YR;
      IF DDISP(DIR#) = 1 THEN DO;
        DATADESC.NUMBYRS_USED = YRCTR;
        RUS_NXT_YR = DATADESC.LSTREC#D;
        RUS_NXT_DISP = DATADESC.LSTDISP + 88;
        IF STRTYR = RUS_YR THEN DO;     END;
          ELSE DO;
            FILYR = STRTYR;
            FIL_NXT_YR = RUS_NXT_YR;
            FIL_NXT_DISP = RUS_NXT_DISP;
            YR_BLK(YRCTR) = FIL_DATA;
            DATADESC.LSTDISP = DATADESC.LSTDISP + 88;
            GO TO PUT_DATA;
            END;
        YR_BLK(YRCTR) = DATA_OUT;
        DATADESC.LSTDISP = DATADESC.LSTDISP + 88;
        END;
      ELSE DO;
        DATADES2.NUMBYRS_USED = YRCTR;
        RUS_NXT_YR = DATADES2.LSTREC#D;
        RUS_NXT_DISP = DATADES2.LSTDISP + 88;
        IF STRTYR = RUS_YR THEN DO;     END;
```

```
            ELSE DO;
                FILYR = STRTYR;
                FIL_NXT_YR = RUS_NXT_YR;
                FIL_NXT_DISP = RUS_NXT_DISP;
                YR_BLK2(YRCTR) = FILDATA;
                DATADES2.LSTDISP = DATADES2.LSTDISP + 88;
                GO TO PUT_DATA;
                END;
            YR_BLK2(YRCTR) = DATA_OUT;
            DATADES2.LSTDISP = DATADES2.LSTDISP + 88;
            END;
        HOLD_YR = INYR;
        RUS_TEMP = -9999;
        RUS_PRECIP = -9999;
        RUS_HARV = 0;
        RUS_PROD = 0;
        GO TO CHK_CDS;
    TERM_ERR:  ;
        PUT SKIP LIST('**CONVERSION ERROR **',INPUT_DATA);
        GO TO EOJ;
    SEQ_ERR:  ;
        PUT SKIP LIST('** DATA NOT IN RIGHT SEQUENCE **',INPUT_DATA);
        GO TO EOJ;
    NO_RM_RUS:  ;
        PUT SKIP LIST('** NOT ENOUGH DISK SPACE ALLOCATED FOR RUSSIA **');
        PUT SKIP LIST(INPUT_DATA);
        PUT SKIP DATA(HOLD_YR,HOLD_LVLCD);
        GO TO EOJ;
    ERRNCD:  ;
        PUT SKIP LIST('** INVALID CODE **',INPUT_DATA);
        GO TO EOJ;
    NEW_STRATA:  ;
        IF YRCTR > 22 THEN GO TO NO_RM_RUS;
        YRCTR = YRCTR + 1;
        RUS_YR = HOLD_YR;
        RUS_NXT_YR = -1;
        RUS_NXT_DISP = 0;
        IF DDISP(DIR#) = 1 THEN DO;
          DATADESC.NUMBYRS_USED = YRCTR;
          YR_BLK(YRCTR) = DATA_OUT;
          END;
        ELSE DO;
          DATADES2.NUMBYRS_USED = YRCTR;
          YR_BLK2(YRCTR) = DATA_OUT;
          END;
        HOLD_YR = INYR;
        HOLD_LVLCD = INLVLCD;
        DIR_CD = HOLD_LVLCD;
        REWRITE FILE(DAF) FROM(DATA_BLK) KEY(DDKEY);
        IF EOF_SW = '1' THEN GO TO EOJ;
        CALL GETDIR(DIR_CD,DIR(*,CONTROL,DAF,DKEY,DIR#,RC));
        DDKEY = DKEY(DIR#);
        READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
        YRCTR = 0;
        STRTYR = 1956;
        RUS_TEMP = -9999;
        RUS_PRECIP = -9999;
        RUS_HARV = 0;
        RUS_PROD = 0;
        GO TO CHK_CDS;
    EOJ:  ;
        FREE RUSSIA_DATA;
        FREE INPUT_DATA;
        FREE FILDATA;
        END RUSSIA;
```

```
RUN NO. 15      DATE  11/12/76    TIME  0910          LISTING OF MODULE AUSARG

DESCRIPTION          LOADS AUSSARG DATA TO DB

MASTER FILE          W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER      11/09/76
LAST DATE COPIED       NONE
LAST UPDATE            NONE

PASSWORD             W267
PROGRAMMER           LEC
LANGUAGE             PLI
PROC PARAMETER       $NOJCL

  //DCOOK JOB ('001000HELHEA  ','COLUM'),COOK,REGION=200K,TIME=1
  //   EXEC SORTD,REGION=200K
  //SORTWK01 DD UNIT=SYSDA,SPACE=(6000,30,,CONTIG)
  //SORTWK02 DD UNIT=SYSDA,SPACE=(6000,30,,CONTIG)
  //SORTWK03 DD UNIT=SYSDA,SPACE=(6000,30,,CONTIG)
  //SORTWK04 DD UNIT=SYSDA,SPACE=(6000,30,,CONTIG)
  //SORTWK05 DD UNIT=SYSDA,SPACE=(6000,30,,CONTIG)
  //SORTWK06 DD UNIT=SYSDA,SPACE=(6000,30,,CONTIG)
  //SORTOUT DD DSN=&&CARDS,DISP=(,PASS),UNIT=SYSDA,SPACE=(1680,(300,30)),
  //   DCB=(RECFM=FB,LRECL=80,BLKSIZE=1680)
  //SORTIN DD DSN=W.EDS.CCEA.MET.AUSDATA,DISP=OLD
  //SYSIN DD *
      SORT FIELDS=(72,9,CH,A,1,10,CH,A),SIZE=E6000
     RECORD TYPE=F,LENGTH=(80)
     END
  /*
  //   EXEC MPL1FCLG
  //PL1L.SYSIN DD *
   ASAR: PROC OPTIONS(MAIN);
      DCL 1 CONTROL,
            2 FILEID CHAR(8),
            2 NUMPASS FIXED BIN(15,0),
            2 PASS(8) CHAR(8),
            2 NUMLEV FIXED BIN(15,0),
            2 LEVNAME(8) CHAR(24),
            2 NUMCODE FIXED BIN(15,0),
            2 CODE(32),
              3 CODENUM FIXED BIN(15,0),
              3 UNITNUM FIXED BIN(15,0),
              3 BASE FIXED BIN(15,0),
              3 SCALE FIXED BIN(15,0),
              3 CODENAME CHAR(24),
              3 UNITNAME CHAR(24),
            2 NUMONE FIXED BIN(15,0),
            2 ONE(24),
              3 CODENUM FIXED BIN(15,0),
              3 NUMDIPS FIXED BIN(15,0),
              3 RECNUM FIXED BIN(15,0),
              3 DISPLACE FIXED BIN(15,0),
              3 NAME CHAR(24),
            2 FILERECS FIXED BIN(15,0),
            2 REC(601),
              3 RECTYPE FIXED BIN(15,0),
              3 FRESPACE FIXED BIN(15,0),
              3 LOCATION FIXED BIN(15,0);
      DCL 1 DIRX,
            2 DIR (44),
            3 LEVNUM FIXED BIN(15,0),
            3 CODENUM FIXED BIN(15,0),
            3 LAT FIXED BIN(15,0),
            3 LON FIXED BIN(15,0),
            3 DIRNAME CHAR(24),
            3 PREC FIXED BIN(15,0),
            3 PDISP FIXED BIN(15,0),
            3 AREC FIXED BIN(15,0),
            3 ADISP FIXED BIN(15,0),
            3 CREC FIXED BIN(15,0),
            3 CDISP FIXED BIN(15,0),
            3 DREC FIXED BIN(15,0),
            3 DDISP FIXED BIN(15,0),
            3 LEVCODE FIXED BIN(31,0),
            3 MODEL(4),
              4 CREC FIXED BIN(15,0),
              4 MREC FIXED BIN(15,0),
              4 MDISP FIXED BIN(15,0),
         2 FILLER CHAR (56);
        DCL 1 INPUT_DATA BASED(J),
              2  INYR          PIC '(7)9',
              2  INCI          PIC '999',
              2  INMON(12)     PIC '$$$$9',
              2  INLVLID,
                 3  INCTRY     PIC '99',
                 3  INREG      PIC '99',
```

```
            3  INZONE    PIC '99',
            3  INSTRA    PIC '99',
            3  INSBSTRA PIC '99';
      DCL 1 DEF_DATA BASED(D),
            2  FIL1          CHAR(10),
            2  TRASDTA(5) PIC '$$$$$$$$9',
            2  INLVLCO PIC '(10)9';
      DCL 1 DEFA_DATA BASED(D),
            2  FIL1          CHAR(10),
            2  ACHAR(12)     CHAR(5),
            2  FIL2          CHAR(10);
      DCL 1 DEFX_DATA BASED(D),
            2  FIL1          CHAR(10),
            2  ACHARX(6)     CHAR(10),
            2  FIL2          CHAR(10);
      DCL 1 ASAR_DATA BASED(P),
            2  ASAR_YR              FIXED BIN(15),
            2  ASAR_NXT_YR          FIXED BIN(15),
            2  ASAR_NXT_DISP        FIXED BIN(15),
            2  RESERV2(17)          FIXED BIN(15),
            2  ASAR_TEMP(12)        FIXED BIN(15),
            2  ASAR_PRECIP(12)      FIXED BIN(15),
            2  ASAR_ZINDEX(12)      FIXED BIN(15),
            2  ASAR_HARV(2)        FIXED BIN(31),
            2  ASAR_PROD(2)         FIXED BIN(31);
      DCL DATA_OUT CHAR(128) BASED(P);
      ALLOCATE ASAR_DATA;
      ASAR_DATA.RESERV2 = 0;
      ALLOCATE INPUT_DATA;
      DCL 1 DATA_BLK,
            2 DATADESC,
              3 ID                  FIXED BIN(31,0),
              3 WMO                 FIXED BIN(31,0),
              3 LATI                FIXED BIN(15,0),
              3 LONGI               FIXED BIN(15,0),
              3 ELEV                FIXED BIN(15,0),
              3 TOTALBLKS_ALLOC FIXED BIN(15,0),
              3 NUMBYTS_USED    FIXED BIN(15,0),
              3 BLOCKSIZE           FIXED BIN(15,0),
              3 FSTRECNO            FIXED BIN(15,0),
              3 FSTDISP             FIXED BIN(15,0),
              3 LSTRECNO            FIXED BIN(15,0),
              3 LSTDISP             FIXED BIN(15,0),
              3 RESERVED            CHAR(18),
              3 NUMBCODE            FIXED BIN(15,0),
              3 DCODE(12),
                4 CODENUMB            FIXED BIN(15,0),
                4 NUMSELEM            FIXED BIN(15,0),
                4 ELEMSIZE            FIXED BIN(15,0),
                4 NUMSCODE            FIXED BIN(15,0),
                4 SUBCODE(8)          FIXED BIN(15,0),
            2  YR_BLK(47)   CHAR(128),
            2  FIL          CHAR(88) INIT(' ');
      DCL HOLD_LVLCO PIC'(10)9';
      DCL HOLD_YR PIC '(7)9';
      DCL DIR_PO FIXED BIN(31);
      DCL FILDATA CHAR(128) BASED(P1);
      DCL 1 DUMASAR BASED(P1),
            2  FILYR FIXED BIN(15),
            2  FIL_NXT_YR FIXED BIN(15),
            2  FIL_NXT_DISP FIXED BIN(15),
            2  FIL_RESV(17) FIXED BIN(15),
            2  FIL_TEMP(12)  FIXED BIN(15),
            2  FIL_PRECIP(12) FIXED BIN(15),
            2  FIL_ZINDEX(12) FIXED BIN(15),
            2  FIL_HARV(2) FIXED BIN(31),
            2  FIL_PROD(2) FIXED BIN(31);
      ALLOCATE FILDATA;
      FIL_RESV = 0;
      FIL_TEMP = -9999;
      FIL_PRECIP = -9999;
      FIL_ZINDEX = -9999;
      FIL_HARV = 0;
      FIL_PROD = 0;
      DCL (DKEY,CKEY,DDKEY,DIR#,RC,YRCTR) FIXED BIN(15);
      CKEY=0;
      DCL STRTYR FIXED BIN(15);
      STRTYR = 1939;
      YRCTR=0;
      RESERV2 = 0;
      ASAR_TEMP = -9999;
      ASAR_PRECIP = -9999;
```

```
      ASAR_ZINDEX = -9999;
      ASAR_HARV = 0;
      ASAR_PROD = 0;
      DCL EOF_SW CHAR(1) INIT('0');
      DCL GETDIR EXTERNAL ENTRY;
      DCL CARDS FILE RECORD INPUT;
      DCL DAF FILE RECORD DIRECT UPDATE KEYED ENV(REGIONAL(1));
      ON CONVERSION GO TO TERM_ERR;
      ON ENDFILE(CARDS) BEGIN;
        EOF_SW = '1';
        GO TO NEW_STRATA;
        END;
FRST_RD:  ;
      READ FILE(CARDS) INTO(INPUT_DATA);
     INYR = TRANSLATE(INYR,'0',' ');
     INCD = TRANSLATE(INCD,'0',' ');
     INLVLCD = TRANSLATE(INLVLCD,'0',' ');
      IF INYR < 1940 THEN GO TO FRST_RD;
      HOLD_YR = INYR;
      HOLD_LVLCD = INLVLCD;
      DIR_CD = HOLD_LVLCD;
      READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
      CALL GETDIR(DIR_CD,DIR(*,CONTROL,DAF,DKEY,DIR#,*),RC);
      IF RC = -1 THEN DO;
        PUT SKIP DATA(DIR_CD,DKEY,DIR#);
        END;
      DDKEY = DREC(DIR#);
      IF DDKEY = -1 THEN DO;
        PUT SKIP DATA(DIR_CD,DKEY,DIR#);
        PUT SKIP LIST(DIR(DIR#));
        PUT SKIP  LIST(INPUT_DATA);
        GO TO EOJ;
        END;
      IF DDKEY < 0 THEN PUT SKIP DATA(DDKEY);
      IF DDKEY > 114 THEN PUT SKIP DATA(DDKEY);
      READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
      GO TO CHK_CDS;
GET_DATA:  ;
      READ FILE(CARDS) INTO(INPUT_DATA);
     INYR = TRANSLATE(INYR,'0',' ');
      IF INYR < 1940 THEN GO TO GET_DATA;
     INCD = TRANSLATE(INCD,'0',' ');
     INLVLCD = TRANSLATE(INLVLCD,'0',' ');
      IF INLVLCD ^= HOLD_LVLCD THEN GO TO NEW_STRATA;
      IF INYR < HOLD_YR THEN GO TO SEQ_ERR;
      IF INYR > HOLD_YR THEN GO TO PUT_DATA;
CHK_CDS:  ;
      IF INCD =  5 THEN GO TO MV_ASAR_PRECIP;
      IF INCD = 35 THEN GO TO MV_ASAR_T;
      IF INCD = 45 THEN GO TO MV_ASAR_ZINDEX;
      IF INCD = 101 THEN GO TO MV_ASAR_HARV;
      /*  IF INCD = 102 THEN GO TO MV_ASAR_PLNT;  */
      IF INCD = 103 THEN GO TO MV_ASAR_PROD;
      GO TO ERRNCD;
MV_ASAR_PRECIP:  ;
      DO N = 1 TO 12;
        IF ACHAR(N) = '    ' THEN INMON(N) = -9999;
        ASAR_PRECIP(N) = INMON(N);
        END;
      GO TO GET_DATA;
MV_ASAR_T:  ;
      DO N = 1 TO 12;
        IF ACHAR(N) = '    ' THEN INMON(N) = -9999;
        ASAR_TEMP(N) = INMON(N);
        END;
      GO TO GET_DATA;
MV_ASAR_ZINDEX:  ;
      DO N = 1 TO 12;
      IF ACHAR(N) = '    ' THEN INMON(N) = -9999;
      ASAR_ZINDEX(N) = INMON(N);
      END;
      GO TO GET_DATA;
MV_ASAR_HARV:  ;
      IF ACHARX(1) = '          ' THEN INAGDTA(1) = -9999;
      IF ACHARX(2) = '          ' THEN INAGDTA(2) = -9999;
      IF DCODE(7).SUBCODE(1) = 201 THEN ASAR_HARV(1) = INAGDTA(1);
      IF DCODE(7).SUBCODE(1) = 202 THEN ASAR_HARV(1) = INAGDTA(1);
      IF DCODE(7).SUBCODE(2) = 202 THEN ASAR_HARV(2) = INAGDTA(2);
      GO TO GET_DATA;
/*  THIS PARAGRAPH IS DUMMIED OUT AS A COMMENT
MV_ASAR_PLNT:  ;
      IF ACHARX(1) = '          ' THEN INAGDTA(1) = -9999;
      IF ACHARX(2) = '          ' THEN INAGDTA(2) = -9999;
      IF DCODE(8).SUBCODE(1) = 201 THEN ASAR_PLNT(1) = INAGDTA(1);
      IF DCODE(8).SUBCODE(1) = 202 THEN ASAR_PLNT(1) = INAGDTA(1);
      IF DCODE(8).SUBCODE(2) = 202 THEN ASAR_PLNT(2) = INAGDTA(2);
      GO TO GET_DATA;
            THE END OF COMMENT       */
```

```
MV_ASAR_PROD:  ;
      IF ACHARX(1) = '                ' THEN INAGDTA(1) = -9999;
      IF ACHARX(2) = '                ' THEN INAGDTA(2) = -9999;
      IF DCODE(8).SUBCODE(1) = 201 THEN ASAR_PROD(1) = INAGDTA(1);
      IF DCODE(8).SUBCODE(1) = 202 THEN ASAR_PROD(1) = INAGDTA(1);
      IF DCODE(8).SUBCODE(2) = 202 THEN ASAR_PROD(2) = INAGDTA(2);
      GO TO GET_DATA;
PUT_DATA:  ;
      IF YRCTR > 47 THEN GO TO NO_RM_ASAR;
      YRCTR = YRCTR + 1;
      STRTYR = STRTYR + 1;
      ASAR_YR = HOLD_YR;
        DATADESC.NUMBYRS_USED = YRCTR;
        ASAR_NXT_YR = DATADESC.LSTRECNO;
        ASAR_NXT_DISP = DATADESC.LSTDISP + 128;
        IF STRTYR = ASAR_YR THEN DO;          END;
          ELSE DO;
            FILYR = STRTYR;
            FIL_NXT_YR = ASAR_NXT_YR;
            FIL_NXT_DISP = ASAR_NXT_DISP;
            YR_BLK(YRCTR) = FILDATA;
            DATADESC.LSTDISP = DATADESC.LSTDISP + 128;
            GO TO PUT_DATA;
            END;
        YR_BLK(YRCTR) = DATA_OUT;
        DATADESC.LSTDISP = DATADESC.LSTDISP + 128;
      HOLD_YR = INYR;
      ASAR_TEMP = -9999;
      ASAR_PRECIP = -9999;
      ASAR_ZINDEX = -9999;
      ASAR_HARV = 0;
      ASAR_PROD = 0;
      GO TO CHK_CDS;
TERM_ERR:  ;
      PUT SKIP LIST('**CONVERSION ERROR **',INPUT_DATA);
      GO TO EOJ;
SEQ_ERR:  ;
      PUT SKIP LIST('** DATA NOT IN RIGHT SEQUENCE **',INPUT_DATA);
      GO TO EOJ;
NO_RM_ASAR:  ;
      PUT SKIP LIST('** NOT ENOUGH DISK SPACE ALLOCATED FOR ASAR **');
      PUT SKIP LIST(INPUT_DATA);
      PUT SKIP DATA(HOLD_YR,HOLD_LVLCD);
      GO TO EOJ;
ERRNCD:  ;
      PUT SKIP LIST('** INVALID CODE **',INPUT_DATA);
      GO TO EOJ;
NEW_STRATA:  ;
      IF YRCTR > 47 THEN GO TO NO_RM_ASAR;
      YRCTR = YRCTR + 1;
      ASAR_YR = HOLD_YR;
      ASAR_NXT_YR = -1;
      ASAR_NXT_DISP = 0;
        DATADESC.NUMBYRS_USED = YRCTR;
        YR_BLK(YRCTR) = DATA_OUT;
      HOLD_YR = INYR;
      HOLD_LVLCD = INLVLCD;
      DIR_CD = HOLD_LVLCD;
      REWRITE FILE(DAF) FROM(DATA_BLK) KEY(DDKEY);
      IF EOF_SW = '1' THEN GO TO EOJ;
      CALL GETDIR(DIR_CD,DIR#,CONTROL,DAF,DKEY,DIR#,RC);
      DDKEY = DREC(DIR#);
      IF DDKEY < 0 THEN DO;
        PUT SKIP DATA(DIR_CD,DIR#,DDKEY);
        PUT SKIP LIST(DIR(DIR#));
        END;
      IF DDKEY > 114 THEN PUT SKIP DATA(DIR_CD,DIR#,DDKEY);
      READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
      YRCTR = 0;
      STRTYR = 1939;
      ASAR_TEMP = -9999;
      ASAR_ZINDEX = -9999;
      ASAR_HARV = 0;
      ASAR_PROD = 0;
      GO TO CHK_CDS;
EOJ:  ;
      FREE ASAR_DATA;
      FREE INPUT_DATA;
      FREE FILDATA;
      END ASAR;
//LKED.SYSLIB DD
//            DD DSN=SYS1.EDS.CCEA.LEC.LOAD,DISP=SHR
//GO.SYSPRINT DD SYSOUT=A
//CARDS DD DSN=&&CARDS,DISP=(OLD,DELETE),UNIT=SYSDA,
//  DCB=(RECFM=FB,LRECL=80,BLKSIZE=1600)
//DAF DD DSN=SYS1.EDS.CCEA.VET.ASSARG,DISP=OLD
/*
```

RUN NO. 15 DATE 11/12/75 TIME 0910 LISTING OF MODULE CANADA

DESCRIPTION DATA BASE PGM

MASTER FILE W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER 10/13/75
LAST DATE COPIED NONE
LAST UPDATE NONE

PASSWORD GOBG
PROGRAMMER LEC
LANGUAGE PLI
PROC PARAMETER $NOJCL

```
CANADA: PROC OPTIONS(MAIN);
   DCL 1 CONTROL,
         2 FILEID CHAR(5),
         2 NUMPASS FIXED BIN(15,0),
         2 PASS(8) CHAR(5),
         2 NUMLEV FIXED BIN(15,0),
         2 LEVNAME(8) CHAR(24),
         2 NUMCODE FIXED BIN(15,0),
         2 CODE(32),
           3 CODENUM FIXED BIN(15,0),
           3 UNITNUM FIXED BIN(15,0),
           3 BASE FIXED BIN(15,0),
           3 SCALE FIXED BIN(15,0),
           3 CODENAME CHAR(24),
           3 UNITNAME CHAR(24),
         2 NUMONE FIXED BIN(15,0),
         2 ONE(24),
           3 CODENUM FIXED BIN(15,0),
           3 NUMDIRS FIXED BIN(15,0),
           3 RECNUM FIXED BIN(15,0),
           3 DISPLACE FIXED BIN(15,0),
           3 NAME CHAR(24),
         2 FILERECS FIXED BIN(15,0),
         2 REC(601),
           3 RECTYPE FIXED BIN(15,0),
           3 FREESPACE FIXED BIN(15,0),
           3 LOCATION FIXED BIN(15,0);
   DCL 1 DIRX,
         2 DIR (84),
           3 LEVNUM FIXED BIN(15,0),
           3 CODENUMB FIXED BIN(15,0),
           3 LAT FIXED BIN(15,0),
           3 LON FIXED BIN(15,0),
           3 DIRNAME CHAR(24),
           3 PDEC FIXED BIN(15,0),
           3 PDISP FIXED BIN(15,0),
           3 BDEC FIXED BIN(15,0),
           3 BDISP FIXED BIN(15,0),
           3 CDEC FIXED BIN(15,0),
           3 CDISP FIXED BIN(15,0),
           3 DDEC FIXED BIN(15,0),
           3 DDISP FIXED BIN(15,0),
           3 LEVCODE FIXED BIN(31,0),
           3 MODEL(4),
             4 CROP FIXED BIN(15,0),
             4 MREC FIXED BIN(15,0),
             4 MDISP FIXED BIN(15,0),
         2 FILLER CHAR (56);
   DCL 1 INPUT_DATA BASED(P),
         2 INYR          PIC '(7)9',
         2 INCD          PIC '999',
         2 INMON(12)     PIC '$$$$9',
         2 INLVLD,
           3 INCTRY      PIC '99',
           3 INREG       PIC '99',
           3 INZONE      PIC '99',
           3 INSTRA      PIC '99',
           3 INSMSTRA    PIC '99';
   DCL 1 DEF_DATA BASED(P),
         2 FIL1          CHAR(10),
         2 INAGDTA(6)    PIC '(10)9',
         2 INLVLCD PIC '(10)9';
   DCL 1 CANADA_DATA BASED(P),
         2 CAN_YR              FIXED BIN(15),
         2 CAN_PRI_YR          FIXED BIN(15),
         2 CAN_NXT_DISP        FIXED BIN(15),
         2 RESERVED            FIXED BIN(15),
         2 CAN_MAX_TEMP(12)    FIXED BIN(15),
         2 CAN_MIN_TEMP(12)    FIXED BIN(15),
         2 CAN_TEMP(12)        FIXED BIN(15),
         2 CAN_PRECIP(12)      FIXED BIN(15),
         2 CAN_PLANT(3)        FIXED BIN(31),
         2 CAN_PROD(3)         FIXED BIN(31);
```

```
      DCL DATA_OUT CHAR(128) BASED(P);
      ALLOCATE CANADA_DATA;
      ALLOCATE INPUT_DATA;
      DCL 1 DATA_BLK,
            2 DATADESC,
              3 ID                  FIXED BIN(31,0),
              3 WMO                 FIXED BIN(31,0),
              3 LATI                FIXED BIN(15,0),
              3 LONGI               FIXED BIN(15,0),
              3 ELEV                FIXED BIN(15,0),
              3 TOTALBLKS_ALLOC     FIXED BIN(15,0),
              3 NUMKEYS_USED        FIXED BIN(15,0),
              3 BLOCKSIZE           FIXED BIN(15,0),
              3 FSTRECNO            FIXED BIN(15,0),
              3 FSTDISP             FIXED BIN(15,0),
              3 LSTRECNO            FIXED BIN(15,0),
              3 LSTDISP             FIXED BIN(15,0),
              3 RESERVED            CHAR(15),
              3 NUMRCODE            FIXED BIN(15,0),
              3 RCODE(12),
                4 CODENUMB            FIXED BIN(15,0),
                4 NUMELEM             FIXED BIN(15,0),
                4 ELEMSIZE            FIXED BIN(15,0),
                4 NUMSCODE            FIXED BIN(15,0),
                4 SUBCODE(8)          FIXED BIN(15,0),
            2 YR_BLK(47)  CHAR(128),
            2 FIL         CHAR(88) INIT(' ');
      DCL HOLD_LVLCD PIC'(10)9';
      DCL HOLD_YR PIC '(7)9';
      DCL DIR_CD FIXED BIN(31);
      DCL (DKEY,CKEY,DDKEY,DIR#,RC,YRCTR) FIXED BIN(15);
      CKEY=0;
      YRCTR=0;
      CAN_MAX_TEMP = -9999;
      CAN_MIN_TEMP = -9999;
      CAN_TEMP     = -9999;
      CAN_PRECIP   = -9999;
      CAN_PLANT    = 0;
      CAN_PROD     = 0;
      DCL EOF_SW CHAR(1) INIT('0');
      DCL GETDIR EXTERNAL ENTRY;
      DCL CARDS FILE RECORD INPUT;
      DCL DAF FILE RECORD DIRECT UPDATE KEYED ENV(REGIONAL(1));
      ON CONVERSION GO TO TERM_ERR;
      ON ENDFILE(CARDS) BEGIN;
        EOF_SW = '1';
        GO TO NEW_STRATA;
        END;
FRST_RD:  ;
      READ FILE(CARDS) INTO(INPUT_DATA);
      INYR = TRANSLATE(INYR,'0',' ');
      INCD = TRANSLATE(INCD,'0',' ');
      INLVLCD = TRANSLATE(INLVLCD,'0',' ');
      HOLD_YR = INYR;
      HOLD_LVLCD = INLVLCD;
      DIR_CD = HOLD_LVLCD;
      READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
      CALL GETDIR(DIR_CD,DIR#,CONTROL,DAF,DKEY,DIR#,RC);
      DDKEY = PREC(DIR#);
      READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
      GO TO CHK_CDS;
GET_DATA:  ;
      READ FILE(CARDS) INTO(INPUT_DATA);
      INYR = TRANSLATE(INYR,'0',' ');
      INCD = TRANSLATE(INCD,'0',' ');
      INLVLCD = TRANSLATE(INLVLCD,'0',' ');
      IF INLVLCD ^= HOLD_LVLCD THEN GO TO NEW_STRATA;
      IF INYR < HOLD_YR THEN GO TO SEQ_ERR;
      IF INYR > HOLD_YR THEN GO TO PUT_DATA;
CHK_CDS:  ;
      IF INCD =   5 THEN GO TO MV_CAN_PRECIP;
      IF INCD =  15 THEN GO TO MV_CAN_TX;
      IF INCD =  25 THEN GO TO MV_CAN_TN;
      IF INCD =  35 THEN GO TO MV_CAN_T;
      IF INCD = 102 THEN GO TO MV_CAN_PLT;
      IF INCD = 103 THEN GO TO MV_CAN_PROD;
      GO TO ERRCD;
MV_CAN_PRECIP:  ;
      DO N = 1 TO 12;
        CAN_PRECIP(N) = INMON(N);
        END;
      GO TO GET_DATA;
```

```
MV_CAN_TX:  ;
      DO N = 1 TO 12;
        CAN_MAX_TEMP(N) = INMON(N);
        END;
      GO TO GET_DATA;
MV_CAN_TN:  ;
      DO N = 1 TO 12;
        CAN_MIN_TEMP(N) = INMON(N);
        END;
      GO TO GET_DATA;
MV_CAN_T:  ;
      DO N = 1 TO 12;
        CAN_TEMP(N) = INMON(N);
        END;
      GO TO GET_DATA;
MV_CAN_PLT:  ;
      CAN_PLANT(1) = INAGDTA(1);
      GO TO GET_DATA;
MV_CAN_PROD:  ;
      CAN_PROD(1) = INAGDTA(1);
      GO TO GET_DATA;
PUT_DATA:  ;
      IF YRCTR > 47 THEN GO TO NO_RM_CAN;
      YRCTR = YRCTR + 1;
      CAN_YR = HOLD_YR;
      LSTRECNO = LSTRECNO;
      LSTDISP = LSTDISP;
      NUMBYRS_USED = YRCTR;
      CAN_NXT_YR = LSTRECNO;
      CAN_NXT_DISP = LSTDISP + 128;
      YR_BLK(YRCTR) = DATA_OUT;
      LSTDISP = LSTDISP + 128;
      HOLD_YR = INYR;
      CAN_MAX_TEMP = -9999;
      CAN_MIN_TEMP = -9999;
      CAN_TEMP     = -9999;
      CAN_PRECIP   = -9999;
      CAN_PLANT    = 0;
      CAN_PROD     = 0;
      GO TO CHK_CDS;
TERM_ERR:  ;
      PUT SKIP LIST('**CONVERSION ERROR **',INPUT_DATA);
      GO TO EOJ;
SEQ_ERR:  ;
      PUT SKIP LIST('** DATA NOT IN RIGHT SEQUENCE **',INPUT_DATA)
      GO TO EOJ;
NO_RM_CAN:  ;
      PUT SKIP LIST('** NOT ENOUGH DISK SPACE ALLOCATED FOR CANADA **');
      GO TO EOJ;
ERRNCD:  ;
      PUT SKIP LIST('** INVALID CODE **',INPUT_DATA);
      GO TO EOJ;
NEW_STRATA:  ;
      IF YRCTR > 47 THEN GO TO NO_RM_CAN;
      YRCTR = YRCTR + 1;
      CAN_YR = HOLD_YR;
      LSTRECNO = LSTRECNO;
      LSTDISP = LSTDISP;
      NUMBYRS_USED = YRCTR;
      CAN_NXT_YR = -1;
      CAN_NXT_DISP = 0;
      YR_BLK(YRCTR) = DATA_OUT;
      HOLD_YR = INYR;
      HOLD_LVLCD = INLVLCD;
      DIR_CD = HOLD_LVLCD;
      REWRITE FILE(DAF) FROM(DATA_BLK) KEY(DDKEY);
      IF EOF_SW = '1' THEN GO TO EOJ;
      CALL GETDIR(DIR_CD,DIRX,CONTROL,DAF,DKEY,DIR#,RC);
      DDKEY = DREC(DIR#);
      READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
      YRCTR = 0;
      CAN_MAX_TEMP = -9999;
      CAN_MIN_TEMP = -9999;
      CAN_TEMP     = -9999;
      CAN_PRECIP   = -9999;
      CAN_PLANT    = 0;
      CAN_PROD     = 0;
      GO TO CHK_CDS;
EOJ:  ;
      FREE CANADA_DATA;
      FREE INPUT_DATA;
      END CANADA;
```

24 75

```
RUN NO. 15      DATE  11/12/75    TIME  0910          LISTING OF MODULE USA

DESCRIPTION          DATA BASE PGM

MASTER FILE          W.EOS.CCEA.LEC.LIBR
ADDED TO MASTER      10/13/75
LAST DATE COPIED       NONE
LAST UPDATE            NONE

PASSWORD             WCEZ
PROGRAMMER           LEC
LANGUAGE             PLI
PROC PARAMETER       $MOJCL


   USA: PROC OPTIONS(MAIN);
      DCL 1 CONTROL,
            2 FILEID CHAR(8),
            2 NUMPASS FIXED BIN(15,0),
            2 PASS(8) CHAR(8),
            2 NUMLEV FIXED BIN(15,0),
            2 LEVNAME(8) CHAR(24),
            2 NUMCODE FIXED BIN(15,0),
            2 CODE(32),
              3 CODENUM FIXED BIN(15,0),
              3 UNITNUM FIXED BIN(15,0),
              3 BASE FIXED BIN(15,0),
              3 SCALE FIXED BIN(15,0),
              3 CODENAME CHAR(24),
              3 UNITNAME CHAR(24),
            2 NUMONE FIXED BIN(15,0),
            2 ONE(24),
              3 CODENUM FIXED BIN(15,0),
              3 NUMDIRS FIXED BIN(15,0),
              3 RECNUM FIXED BIN(15,0),
              3 DISPLACE FIXED BIN(15,0),
              3 NAME CHAR(24),
            2 FILESPECS FIXED BIN(15,0),
            2 REC(601),
              3 RECTYPE FIXED BIN(15,0),
              3 FREESPACE FIXED BIN(15,0),
              3 LOCATION FIXED BIN(15,0);
      DCL 1 DIRX,
            2 DIR (64),
              3 LEVNUM FIXED BIN(15,0),
              3 CODENUM FIXED BIN(15,0),
              3 LAT FIXED BIN(15,0),
              3 LON FIXED BIN(15,0),
              3 DIRNAME CHAR(24),
              3 PREC FIXED BIN(15,0),
              3 PDISP FIXED BIN(15,0),
              3 SREC FIXED BIN(15,0),
              3 SDISP FIXED BIN(15,0),
              3 CREC FIXED BIN(15,0),
              3 CDISP FIXED BIN(15,0),
              3 DREC FIXED BIN(15,0),
              3 DDISP FIXED BIN(15,0),
              3 LEVCODE FIXED BIN(31,0),
              3 MODEL(4),
                4 CROP FIXED BIN(15,0),
                4 MREC FIXED BIN(15,0),
                4 MDISP FIXED BIN(15,0),
            2 FILLER CHAR (56);
       DCL 1 INPUT_DATA BASED(0),
             2  INYR          PIC '(7)9',
             2  INCD          PIC '999',
             2  INMON(12)     PIC '$$$$9',
             2  INLVLID,
                3  INCTRY     PIC '99',
                3  INREG      PIC '99',
                3  INZONE     PIC '99',
                3  INSTRA     PIC '99',
                3  INSBSTRA   PIC '99';
       DCL 1 DEF_DATA BASED(0),
             2  FIL1          CHAR(10),
             2  INAGDTA(6)    PIC '$$$$$$$$$9',
             2  INLVLCD PIC '(10)9';
       DCL 1 DEFA_DATA BASED(),
             2  FIC1          CHAR(10),
             2  ACHAR(12)     CHAR(5),
             2  FIL2          CHAR(10);
       DCL 1 DEFX_DATA BASED(),
             2  FIC1          CHAR(10),
             2  ACHARX(6)     CHAR(10),
             2  FIL2          CHAR(10);
```

3-27 #76

```
      DCL 1 USA_DATA BASED(P),
            2  USA_YR                FIXED BIN(15),
            2  USA_NXT_YR             FIXED BIN(15),
            2  USA_NXT_DISP           FIXED BIN(15),
            2  RESERV2                FIXED BIN(15),
            2  USA_TEMP(12)           FIXED BIN(15),
            2  USA_PRECIP(12)         FIXED BIN(15),
            2  USA_DEGDAY(12)         FIXED BIN(15),
            2  USA_HARV(4)            FIXED BIN(31),
            2  USA_PLNT(4)            FIXED BIN(31),
            2  USA_PROD(4)            FIXED BIN(31);
      DCL DATA_OUT CHAR(128) BASED(P);
      ALLOCATE USA_DATA;
      USA_DATA.RESERV2 = 0;
      ALLOCATE INPUT_DATA;
      DCL 1 DATA_BLK=
            2  DATADESC,
               3 ID                   FIXED BIN(31,0),
               3 YMO                  FIXED BIN(31,0),
               3 LATI                 FIXED BIN(15,0),
               3 LONGI                FIXED BIN(15,0),
               3 ELEV                 FIXED BIN(15,0),
               3 TOTALBLKS_ALLOC      FIXED BIN(15,0),
               3 NUMBYRS_USED         FIXED BIN(15,0),
               3 BLOCKSIZE            FIXED BIN(15,0),
               3 FSTRECNO             FIXED BIN(15,0),
               3 FSTDISP              FIXED BIN(15,0),
               3 LSTRECNO             FIXED BIN(15,0),
               3 LSTDISP              FIXED BIN(15,0),
               3 RESERVED             CHAR(16),
               3 NUMCODE              FIXED BIN(15,0),
               3 DCODE(12),
                 4 CODENUM              FIXED BIN(15,0),
                 4 NUMSELEM             FIXED BIN(15,0),
                 4 ELEMSIZE             FIXED BIN(15,0),
                 4 NUMSCODE             FIXED BIN(15,0),
                 4 SUBCODE(4)           FIXED BIN(15,0),
            2  YR_BLK(47)  CHAR(128),
            2  FIL                    CHAR(64) INIT(' ');
      DCL HOLD_LALCD PIC'(10)9';
      DCL HOLD_YR PIC '(7)9';
      DCL DIR_CD FIXED BIN(31);
      DCL FILDATA CHAR(128) BASED(P1);
      DCL 1 DUMUSA BASED(P1),
            2  FILYR FIXED BIN(15),
            2  FIL_NXT_YR FIXED BIN(15),
            2  FIL_NXT_DISP FIXED BIN(15),
            2  FIL_RESV FIXED BIN(15),
            2  FIL_TEMP(12)   FIXED BIN(15),
            2  FIL_PRECIP(12) FIXED BIN(15),
            2  FIL_DEGDAY(12)  FIXED BIN(15),
            2  FIL_HARV(4) FIXED BIN(31),
            2  FIL_PLNT(4)     FIXED BIN(31),
            2  FIL_PROD(4) FIXED BIN(31);
      ALLOCATE FILDATA;
      FIL_RESV = 0;
      FIL_TEMP = -9999;
      FIL_PRECIP = -9999;
        FIL_DEGDAY = -9999;
      FIL_HARV = 0;
       FIL_PLNT = 0;
      FIL_PROD = 0;
      DCL (DKEY,CKEY,OKEY,DIR#,PC,YRCTR) FIXED BIN(15);
      CKEY=0;
      DCL STRTYR FIXED BIN(15);
      STRTYR = 1930;
      YRCTR=0;
      RESERV2 = 0;
      USA_TEMP = -9999;
      USA_PRECIP = -9999;
      USA_DEGDAY = -9999;
      USA_HARV = 0;
      USA_PLNT = 0;
      USA_PROD = 0;
      DCL EOF_S  CHAR(1) INIT('0');
      DCL GETDIR EXTERNAL ENTRY;
      DCL CARDS FILE RECORD INPUT;
      DCL DAF FILE RECORD DIRECT UPDATE KEYED ENV(REGIONAL(1));
      ON CONVERSION GO TO TERM_ERR;
      ON ENDFILE(CARDS) BEGIN;
        EOF_S = '1';
        GO TO NEW_STRATA;
        END;
   FRST_RD:   ;
      READ FILE(CARDS) INTO(INPUT_DATA);
```

#77

```
      INYR = TRANSLATE(INYR,'0',' ');
      INCD = TRANSLATE(INCD,'0',' ');
      INLVLCD = TRANSLATE(INLVLCD,'0',' ');
       IF INYR < 1931 THEN GO TO FRST_RD;
       HOLD_YR = INYR;
       HOLD_LVLCD = INLVLCD;
       DIR_CD = HOLD_LVLCD;
       READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
       CALL GETDIR(DIR_CD,DIRX,CONTROL,DAF,DKEY,DIR#,RC);
       IF RC = -1 THEN DO;
         PUT SKIP DATA(DIR_CD,DKEY,DIR#);
         END;
       DDKEY = DREC(DIR#);
       IF DDKEY = -1 THEN DO;
         PUT SKIP DATA(DIR_CD,DKEY,DIR#);
         PUT SKIP LIST(DIRTDIR#));
         PUT SKIP  LIST(INPUT_DATA);
         GO TO EOJ;
         END;
       IF DDKEY < 0 THEN PUT SKIP DATA(DDKEY);
       IF DDKEY > 114 THEN PUT SKIP DATA(DDKEY);
       READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
       GO TO CHK_CDS;
  GET_DATA:  ;
       READ FILE(CARDS) INTO(INPUT_DATA);
      INYR = TRANSLATE(INYR,'0',' ');
        IF INYR < 1931 THEN GO TO GET_DATA;
      INCD = TRANSLATE(INCD,'0',' ');
      INLVLCD = TRANSLATE(INLVLCD,'0',' ');
       IF INLVLCD = 030140000 THEN GO TO GET_DATA;
       IF INLVLCD ¬= HOLD_LVLCD THEN GO TO NEW_STRATA;
       IF INYR < HOLD_YR THEN GO TO SEQ_ERR;
       IF INYR > HOLD_YR THEN GO TO PUT_DATA;
  CHK_CDS:  ;
       IF INCD =   5 THEN GO TO MV_USA_PRECIP;
       IF INCD =  35 THEN GO TO MV_USA_T;
       IF INCD = 40 THEN GO TO MV_USA_DEGDAY;
       IF INCD = 101 THEN GO TO MV_USA_HARV;
       IF INCD = 102 THEN GO TO MV_USA_PLNT;
       IF INCD = 103 THEN GO TO MV_USA_PROD;
       GO TO ERRNCD;
  MV_USA_PRECIP:  ;
       DO N = 1 TO 12;
         IF ACHAR(N) = '      ' THEN INMON(N) = -9999;
         USA_PRECIP(N) = INMON(N);
         END;
       GO TO GET_DATA;
  MV_USA_T:  ;
       DO N = 1 TO 12;
         IF ACHAR(N) = '      ' THEN INMON(N) = -9999;
         USA_TEMP(N) = INMON(N);
         END;
       GO TO GET_DATA;
  MV_USA_DEGDAY:  ;
       DO N = 1 TO 12;
       IF ACHAR(N) = '      ' THEN INMON(N) = -9999;
       USA_DEGDAY(N) = INMON(N);
       END;
       GO TO GET_DATA;
  MV_USA_HARV:  ;
       IF ACHARX(1) = '          ' THEN INAGDTA(1) = -9999;
       IF ACHARX(2) = '          ' THEN INAGDTA(2) = -9999;
       IF DCODE(7).SUBCODE(1) = 201 THEN USA_HARV(1) = INAGDTA(1);
       IF DCODE(7).SUBCODE(1) = 202 THEN USA_HARV(1) = INAGDTA(1);
       IF DCODE(7).SUBCODE(2) = 202 THEN USA_HARV(2) = INAGDTA(2);
       GO TO GET_DATA;
  MV_USA_PLNT:  ;
       IF ACHARX(1) = '          ' THEN INAGDTA(1) = -9999;
       IF ACHARX(2) = '          ' THEN INAGDTA(2) = -9999;
       IF DCODE(8).SUBCODE(1) = 201 THEN USA_PLNT(1) = INAGDTA(1);
       IF DCODE(8).SUBCODE(1) = 202 THEN USA_PLNT(1) = INAGDTA(1);
       IF DCODE(8).SUBCODE(2) = 202 THEN USA_PLNT(2) = INAGDTA(2);
       GO TO GET_DATA;
  MV_USA_PROD:  ;
       IF ACHARX(1) = '          ' THEN INAGDTA(1) = -9999;
       IF ACHARX(2) = '          ' THEN INAGDTA(2) = -9999;
       IF DCODE(9).SUBCODE(1) = 201 THEN USA_PROD(1) = INAGDTA(1);
       IF DCODE(9).SUBCODE(1) = 202 THEN USA_PROD(1) = INAGDTA(1);
       IF DCODE(9).SUBCODE(2) = 202 THEN USA_PROD(2) = INAGDTA(2);
       GO TO GET_DATA;
  PUT_DATA:  ;
       IF YRCTR > 47 THEN GO TO TO_RM_USA;
       YRCTR = YRCTR + 1;
```

```
      GO TO EOJ;
NO_RM_USA:  :
      PUT SKIP LIST('** NOT ENOUGH DISK SPACE ALLOCATED FOR USA **');
      PUT SKIP LIST(INPUT_DATA);
      PUT SKIP DATA(HOLD_YR,HOLD_LVLCD);
      GO TO EOJ;
ERRNCD:  :
      PUT SKIP LIST('** INVALID CODE **',INPUT_DATA);
      GO TO EOJ;
NEW_STRATA:  :
      IF YRCTR > 47 THEN GO TO NO_RM_USA;
      YRCTR = YRCTR + 1;
      USA_YR = HOLD_YR;
      USA_NXT_YR = -1;
      USA_NXT_DISP = 0;
        DATADESC.NUMBYRS_USED = YRCTR;
        YR_BLK(YRCTR) = DATA_OUT;
      HOLD_YR = INYR;
      HOLD_LVLCD = INLVLCD;
      DIR_CD = HOLD_LVLCD;
      REWRITE FILE(DAF) FROM(DATA_BLK) KEY(DDKEY);
      IF EOF_SW = '1' THEN GO TO EOJ;
      CALL GETDIR(DIR_CD,DIR#,CONTROL,DAF,DKEY,DIR#,RC);
      DDKEY = DREC(DIR#);
      IF DDKEY < 0 THEN DO;
        PUT SKIP DATA(DIR_CD,DIR#,DDKEY);
        PUT SKIP LIST(DIR#DIR#);
        END;
      IF DDKEY > 114 THEN PUT SKIP DATA(DIR_CD,DIR#,DDKEY);
      READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
      YRCTR = 0;
      STRTYR = 1930;
      USA_TEMP = -9999;
      USA_PRECIP = -9999;
      USA_DEGDAY = -9999;
      USA_MARV = 0;
      USA_PLNT = 0;
      USA_PROD = 0;
      GO TO CHK_CDS;
EOJ:  :
      FREE USA_DATA;
      FREE INPUT_DATA;
      FREE FILDATA;
      END USA;
      STRTYR = STRTYR + 1;
      USA_YR = HOLD_YR;
        DATADESC.NUMBYRS_USED = YRCTR;
        USA_NXT_YR = DATADESC.LSTPRCYR;
        USA_NXT_DISP = DATADESC.LSTDISP + 128;
        IF STRTYR = USA_YR THEN DO;        END;
          ELSE DO;
            FILYR = STRTYR;
            FIL_NXT_YR = USA_NXT_YR;
            FIL_NXT_DISP = USA_NXT_DISP;
            YR_BLK(YRCTR) = FILDATA;
            DATADESC.LSTDISP = DATADESC.LSTDISP + 128;
            GO TO PUT_DATA;
            END;
        YR_BLK(YRCTR) = DATA_OUT;
        DATADESC.LSTDISP = DATADESC.LSTDISP + 128;
      HOLD_YR = INYR;
      USA_TEMP = -9999;
      USA_PRECIP = -9999;
      USA_DEGDAY = -9999;
      USA_MARV = 0;
      USA_PLNT = 0;
      USA_PROD = 0;
      GO TO CHK_CDS;
TERM_ERR:  :
      PUT SKIP LIST('**CONVERSION ERROR **',INPUT_DATA);
      GO TO EOJ;
SEQ_ERR:  :
      PUT SKIP LIST('** DATA NOT IN RIGHT SEQUENCE **',INPUT_DATA);
```

3.3.13 CONTROL BLOCK LISTER (YESLS02)

YESLS02 is provided to list the contents of the control block.

3.3.13.1 Linkages

None.

3.3.13.2 Interfaces

INITIAL must be run before YESLS02.

3.3.13.3 Inputs

Card containing ++END OF COMMAND.

3.3.13.4 Outputs

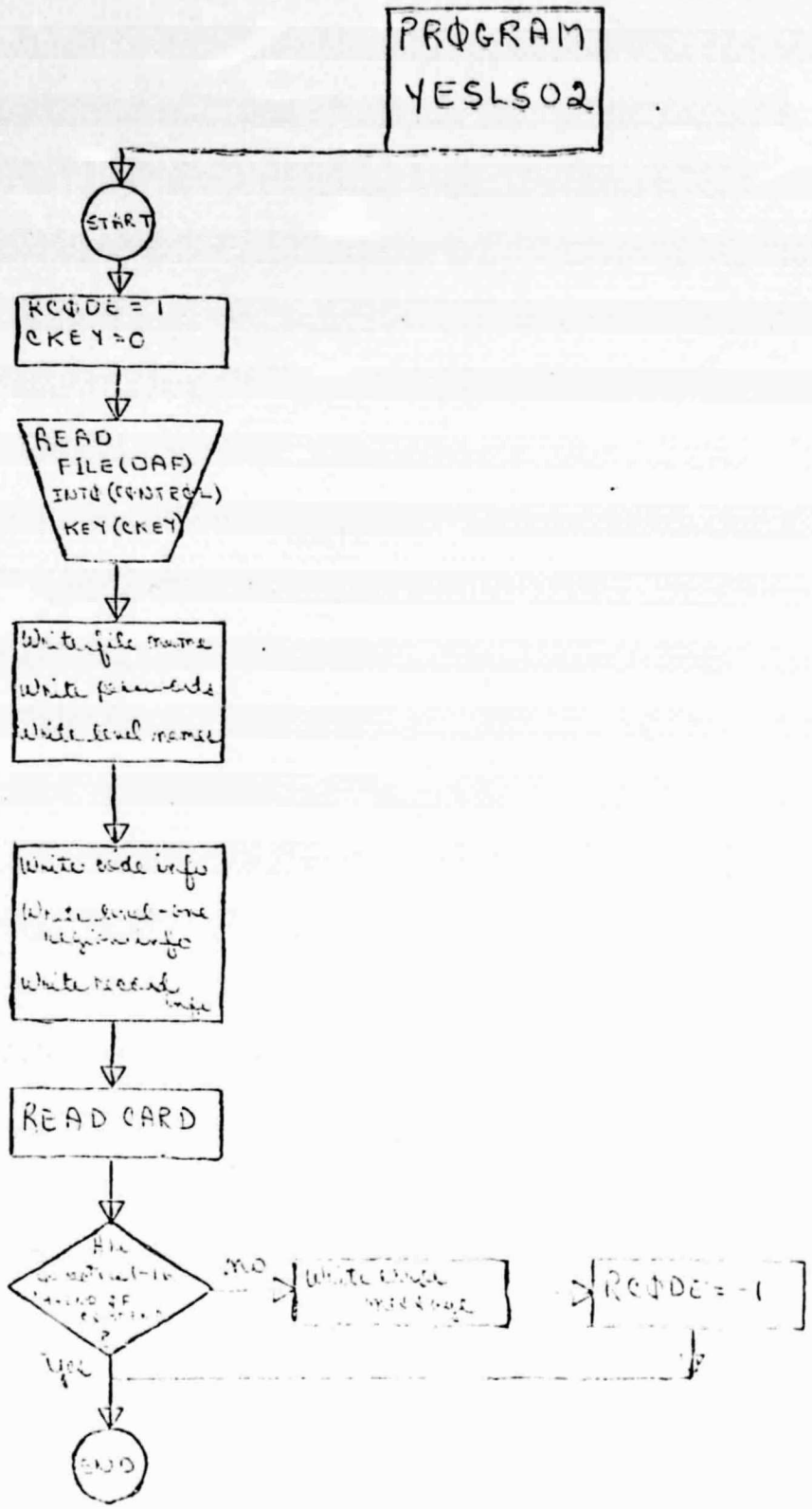
Control block listing.

3.3.13.5 Flow Chart

Next page.

3.3.13.6 Listing

Follows flow chart.


PROGRAM
YESLS02
START
RCODE = 1
CKEY = 0
READ
FILE(DAF)
INTO (CONTROL)
KEY(CKEY)
Write file name
Write parameters
Write level names
Write code info
Write level - one
Write record
READ CARD
NO
RCODE = -1
YES
END

3-79 80 81

```
RUN NO. 15        DATE 11/12/76      TIME  0910              LISTING OF MODULE LISTCON

DESCRIPTION             LIST DATA BASE PGM

MASTER FILE             W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER         10/13/76
LAST DATE COPIED          NONE
LAST UPDATE               NONE

PASSWORD                VKRH
PROGRAMMER              LEC
LANGUAGE                PLI
PROC PARAMETER          $NOJCL

   YESLS02: PROC OPTIONS(MAIN);
   /*                                                                    */
   /* T[illegible] TO LIST [illegible] */
   /*                                                                    */
    DCL 1 CONTROL,
            2 FILEID CHAR(8),
            2 NUMPASS FIXED BIN(15,0),
            2 PASS(8) CHAR (8),
            2 NUMLEV FIXED BIN(15,0),
            2 LEVNAME(8) CHAR(24),
            2 NUMCODE FIXED BIN(15,0),
            2 CODE(32),
              3 CODENUM FIXED BIN(15,0),
              3 UNITNUM FIXED BIN(15,0),
              3 BASE FIXED BIN(15,0),
              3 SCALE FIXED BIN(15,0),
              3 CODENAME CHAR(24),
              3 UNITNAME CHAR(24),
            2 NUMONE FIXED BIN(15,0),
            2 ONE(24),
              3 CODEINUM FIXED BIN(15,0),
              3 NUMDIRS FIXED BIN(15,0),
              3 RECNUM FIXED BIN(15,0),
              3 DISPLACE FIXED BIN(15,0),
              3 NAME CHAR(24),
            2 FILERECS FIXED BIN(15,0),
            2 REC(601),
              3 RECTYPE FIXED BIN(15,0),
              3 FRESPACE FIXED BIN(15,0),
              3 LOCATION FIXED BIN(15,0);
    DCL CKEY FIXED BIN(10,0);
    DCL P1 POINTER;
    DCL D CHAR(8) BASED(P1);
    DCL B BIT(64) BASED(P1);
    DCL (I,J,NJOB,RCODE) FIXED BIN(15,0);
    DCL INSTR CHAR(80);
    DCL SYSIN FILE STREAM INPUT;
    DCL SYSPRINT FILE STREAM OUTPUT;
    DCL DAF FILE RECORD DIRECT KEYED ENV(REGIONAL(1));
    DCL SUBSTR BUILTIN;
   DCL PW(8) CHAR(8);
   DCL CONCARD CHAR(16);
    OPEN FILE(SYSPRINT) STREAM OUTPUT LINESIZE(130);
```

ORIGINAL PAGE IS
OF POOR QUALITY

3-80

81 82

```
   RCODE=1;
   CKEY,I=0;
   READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
   PUT SKIP FILE(SYSPRINT) EDIT('***LIST CONTROL BLOCK PROGRAM***',
     'THE FILE IDENTIFICATION NAME IS ',FILEID) (A,SKIP(4),A,A(8));
   ALLOCATE D SET(P1);
   DO I=1 TO NUMPASS;
     D=PASS(I);
           B=¬B;
     PW(I)=D;
   END;
   PUT SKIP(3) FILE(SYSPRINT) EDIT ('THE PASSWORD(S) ARE ',(PW(I) DO
     I=1 TO NUMPASS)) (A,8 A);
   PUT SKIP(3) FILE(SYSPRINT) EDIT ('THE LEVEL NAMES ARE ',(LEVNAME(I)
     DO I=1 TO NUMLEV)) (A,4 A, SKIP, X(20),4 A);
   PUT SKIP(3) FILE(SYSPRINT) EDIT
     ('THE FOLLOWING CODE NUMBERS ARE USED FOR DATA IN THE FILE',
     'CODE#  UNIT#  CODE NAME','UNIT NAME','BASE  SCALE',
     ( CODE(I),CODENUM,CODE(I),UNITNUM,CODE(I),CODENAME,CODE(I),UNITNAME,
     CODE(I),BASE,CODE(I),SCALE  DO I=1 TO NUMCODE)) (A,SKIP,X(5),A,
     2(X(15),A),SKIP,32(2 F(7,0),X(3),2 A(24),F(4,0),F(6,0),SKIP));
   PUT SKIP(3) FILE(SYSPRINT) EDIT
     ('THE FOLLOWING COUNTRIES ARE INCLUDED IN THE FILE','CODE#  COUNTRY',
     'NUMBER OF DIRECTORIES  LOCATION OF FIRST DIRECTORY',
     ( ONE(I),CODENUM,ONE(I),NAME,ONE(I),NUMDIRS,ONE(I),RECNUM,
     ONE(I),DISPLACE  DO I=1 TO NUMONE)) (A,SKIP,X(3),A,X(18),A,SKIP,
     24(F(7,0),X(3),A,X(9),F(4,0),X(18),2 F(5,0),SKIP));
   PUT SKIP(3) FILE(SYSPRINT) EDIT('THE FILE HAS BEEN DEFINED TO CONTAIN',
     FILERECS,' RECORDS, EACH 6440 BYTES LONG',
     'RECORD NUMBER ZERO CONTAINS THE CONTROL BLOCK FOR THE FILE',
     'KEY TO RECORD TYPE CODES: CODE#  RECORD TYPE',' 0    BLANK,UNUSED',
     '-1    DIRECTORY BLOCK','+1    DATA DESCRIPTOR AND DATA BLOCK',
     '+2    MODEL DEFINITION BLOCK') (A,F(4,0),A,SKIP(3),A,SKIP(3),
     A,SKIP,4(X(27),A,SKIP));
   PUT PAGE FILE(SYSPRINT) EDIT
     ('RECORD#  TYPE  FREESPACE(IN BYTES)  LOCATION OF FREESPACE',
      'RECORD#  TYPE  FREESPACE(IN BYTES)  LOCATION OF FREESPACE',
     ( T,REC(I),RECTYPE,REC(I),FREESPACE,REC(I),LOCATION  DO I=1 TO
     FILERECS)) (X(2),A,X(11),A,SKIP,300(2 F(7,0),X(9),F(4,0),X(18),
     F(4,0),X(19),2 F(7,0),X(4),F(4,0),X(19),F(4,0),SKIP));
   GET FILE(SYSIN) EDIT (CONCARD) (COL(1),A(15));
   IF CONCARD='***END OF COMMAND' THEN GOTO EOJ;
   PUT SKIP FILE(SYSPRINT) EDIT ('***END OF COMMAND CARD MISSING***') (A);
   RCODE=-1;
   FREE D;
   EOJ:RETURN;
   END YESLS02;
```

~~3-81~~

82 83

3.3.14 DIRECTORY BLOCK LISTER (YESLS04)

YESLS04 is provided to list directory information.

3.3.14.1 Linkages

None.

3.3.14.2 Interfaces

The directories requested must have been defined.

3.3.14.3 Inputs

Cards requesting directories.

3.3.14.4 Outputs

Directory listings.

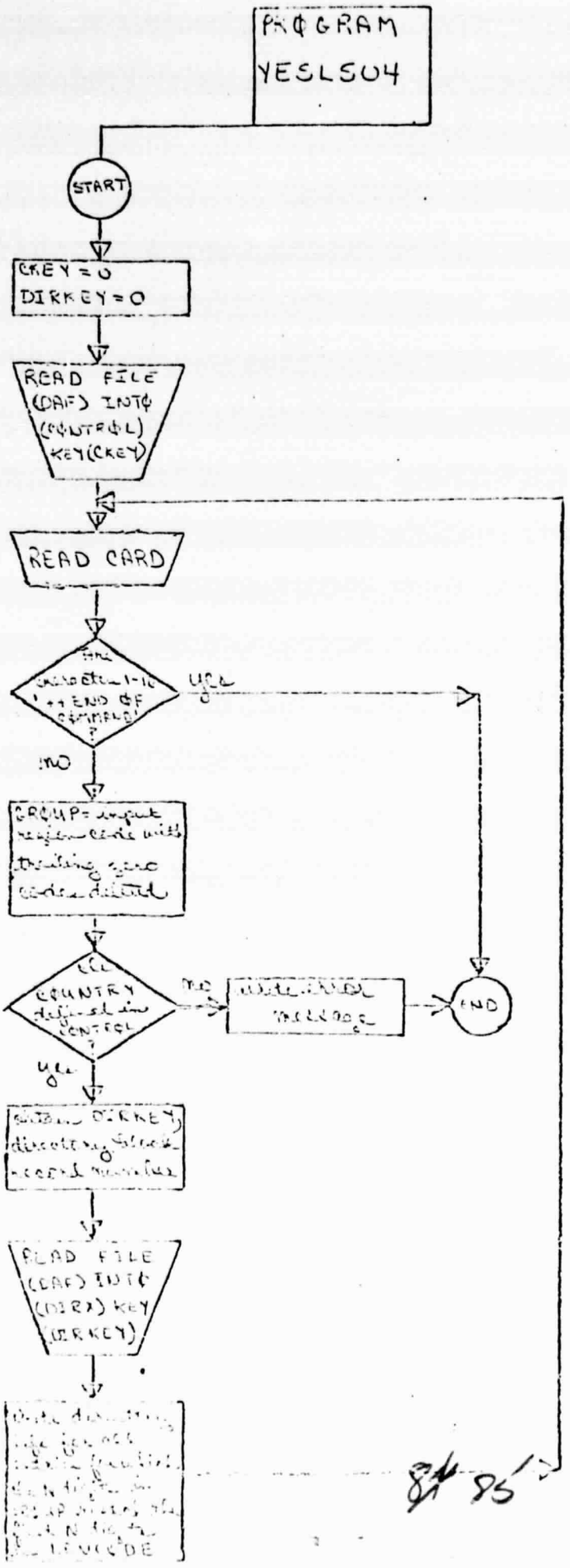
3.3.14.5 Flow Chart

Next page.

3.3.14.6 Listing

Follows flow chart.

8384


PROGRAM
YESLS04
START
CKEY = 0
DIRKEY = 0
READ FILE
(DAF) INTO
(COUNTROL)
KEY(CKEY)
READ CARD
YES
NO
GROUP = input
region code with
trailing zero
codes deleted
YES
COUNTRY
defined in
CONTROL
?
NO
END
YES
NO
8 4 85

```
RUN NO. 18      DATE  11/12/75    TIME  0920          LISTING OF MODULE LISTDIR

DESCRIPTION          LIST DATA BASE PGM

MASTER FILE          W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER      10/13/75
LAST DATE COPIED       NONE
LAST UPDATE            NONE

PASSWORD             XGHT
PROGRAMMER           LEC
LANGUAGE             PLI
PROC PARAMETER       $NOJCL

YESLS04: PROC OPTIONS(MAIN);
/*                                                              */
/*THIS PROGRAM IS CALLED [illegible] LIST DIRECTORY - YES*/
/*                                                              */
   DCL 1 CONTROL,
         2 FILEID CHAR(8),
         2 NUMPASS FIXED BIN(15,0),
         2 PASS(8) CHAR(8),
         2 NUMLEV FIXED BIN(15,0),
         2 LEVNAME(8) CHAR(24),
         2 NUMCODE FIXED BIN(15,0),
         2 CODE(32),
           3 CODENUM FIXED BIN(15,0),
           3 UNITNUM FIXED BIN(15,0),
           3 BASE FIXED BIN(15,0),
           3 SCALE FIXED BIN(15,0),
           3 CODENAME CHAR(24),
           3 UNITNAME CHAR(24),
         2 NUMONE FIXED BIN(15,0),
         2 ONE(24),
           3 CODENUM FIXED BIN(15,0),
           3 NUMDIRS FIXED BIN(15,0),
           3 RECNUM FIXED BIN(15,0),
           3 DISPLACE FIXED BIN(15,0),
           3 NAME CHAR(24),
         2 FILERECS FIXED BIN(15,0),
         2 REC(601),
           3 RECTYPE FIXED BIN(15,0),
           3 FREESPACE FIXED BIN(15,0),
           3 LOCATION FIXED BIN(15,0);
   DCL 1 DIRX,
         2 DIR (84),
           3 LEVNUM FIXED BIN(15,0),
           3 CODENUMB FIXED BIN(15,0),
           3 LAT FIXED BIN(15,0),
           3 LON FIXED BIN(15,0),
           3 DIRNAME CHAR(24),
           3 PREC FIXED BIN(15,0),
           3 PDISP FIXED BIN(15,0),
           3 BREC FIXED BIN(15,0),
           3 BDISP FIXED BIN(15,0),
           3 CREC FIXED BIN(15,0),
           3 CDISP FIXED BIN(15,0),
```

```
          3 DREC FIXED BIN(15,0),
          3 DDISP FIXED BIN(15,0),
          3 LEVCODE FIXED BIN(31,0),
          3 MODEL(4),
            4 CREC FIXED BIN(15,0),
            4 MREC FIXED BIN(15,0),
            4 MDISP FIXED BIN(15,0),
        2 FILLER CHAR (56);
  DCL (CKEY,DIRKEY,DIRNUM,I,K,M,COUNTRY) FIXED BIN(15,0);
  DCL SYSIN FILE STREAM INPUT;
  DCL SYSPRINT FILE STREAM OUTPUT;
  DCL DAF FILE RECORD DIRECT KEYED ENV(REGIONAL(1));
  DCL INSTR CHAR(80);
  DCL SUBSTR BUILTIN;
  DCL (CODES,GROUP,DIV) FIXED BIN(31,0);
   OPEN FILE(SYSPRINT) STREAM OUTPUT LINESIZE(130);
  ON ENDFILE(SYSIN) BEGIN;
    PUT SKIP FILE(SYSPRINT) EDIT
      ('***ERROR - END OF FILE SYSIN ENCOUNTERED***') (A);
    RCODE=-1;
    GOTO EOJ;
  END;
  PUT SKIP FILE(SYSPRINT) EDIT('***LIST DIRECTORY PROGRAM***') (A);
  RCODE=1;
  CKEY=0;
  READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
  GETCARD: GET FILE(SYSIN) EDIT(INSTR) (COL(1),A(80));
  IF SUBSTR(INSTR,1,16)='**END OF COMMAND' THEN GOTO EOJ;
    DIV=1;
    IF SUBSTR(INSTR,10,2)='00' THEN DIV=100;
    IF SUBSTR(INSTR,8,4)='0000' THEN DIV=10000;
    IF SUBSTR(INSTR,6,6)='000000' THEN DIV=1000000;
    IF SUBSTR(INSTR,4,8)='00000000'THEN DIV=100000000;
  GET STRING(INSTR) EDIT (CODES) (X(1),F(10,0));
  GET STRING(INSTR) EDIT (COUNTRY) (X(1),F(2,0));
  DIRKEY,DIRNUM,I,K,M=0;
  GROUP=CODES/DIV;
  DO I = 1 TO CONTROL.NUMONE;
    IF COUNTRY=CONTROL.ONE(I).CODE1NUM THEN DO;
      DIRKEY=CONTROL.ONE(I).RECNUM;
      DIRNUM=CONTROL.ONE(I).NUMDIRS;
      PUT PAGE FILE(SYSPRINT) EDIT (' THE COUNTRY ',CONTROL.ONE(I).NAME,
        'HAS ',CONTROL.ONE(I).NUMDIRS,' DIRECTORY ENTRIES ON RECORD ',
        DIRKEY) (A,A(24),A,F(4,0),A,F(4,0));
    END;
  END;
  IF DIRKEY>0 THEN DO;
      PUT SKIP(4) FILE(SYSPRINT) EDIT ('LEVEL','CODE','NAME','LATITUDE'
        ,'LONGITUDE','PARENT','BROTHER','CHILD','DATA','MODELS')
        (A,X(1),A,X(2),A,X(19),A,X(1),A,X(2),A,X(4),A,X(2),A,X(5),A,
        X(5),A);
    READ FILE(DAF) INTO(DIRX) KEY(DIRKEY);
    DO K = 1 TO DIRNUM;
      M=M+1;
      IF M=85 THEN DO;
        M=1;
        DIRKEY=DIRKEY+1;
        READ FILE(DAF) INTO(DIRX) KEY(DIRKEY);
      END;
     IF GROUP=FLOOR(DIR(M).LEVCODE/DIV) THEN PUT SKIP FILE(SYSPRINT) EDIT
        (DIR(M).LEVNUM,DIR(M).CODENUMB,DIR(M).DIRNAME,DIR(M).LAT,
        DIR(M).LON,DIR(M).PREC,DIR(M).PDISP,DIR(M).SREC,DIR(M).SDISP,
        DIR(M).CREC,DIR(M).CDISP,DIR(M).DREC,DIR(M).DDISP,
        ( DIR(M).MODEL(J).MREC,DIR(M).MODEL(J).MDISP  DO J=1 TO 4))
        (F(3,0),F(5,0),X(4),A(24),P'---9.9',X(3),P'---9.9',
        8 F(5,0),X(1),4(F(4,0),F(5,0)));
    END;
  END;
  ELSE PUT SKIP FILE(SYSPRINT)
    EDIT('COUNTRY HAS NO DIRECTORY ENTRY') (A);
  GOTO GETCARD;
  EOJ: END YESLS04;
```

3.3.15 LISTING DATA IN THE DATA BASE (LISTJOB)

LISTJOB is provided to list data in the data base.

3.3.15.1 Linkages

None.

3.3.15.2 Interfaces

Data and control block entries must exist for the countries requested.

3.3.15.3 Inputs

Request for data cards on a country basis.

3.3.15.4 Outputs

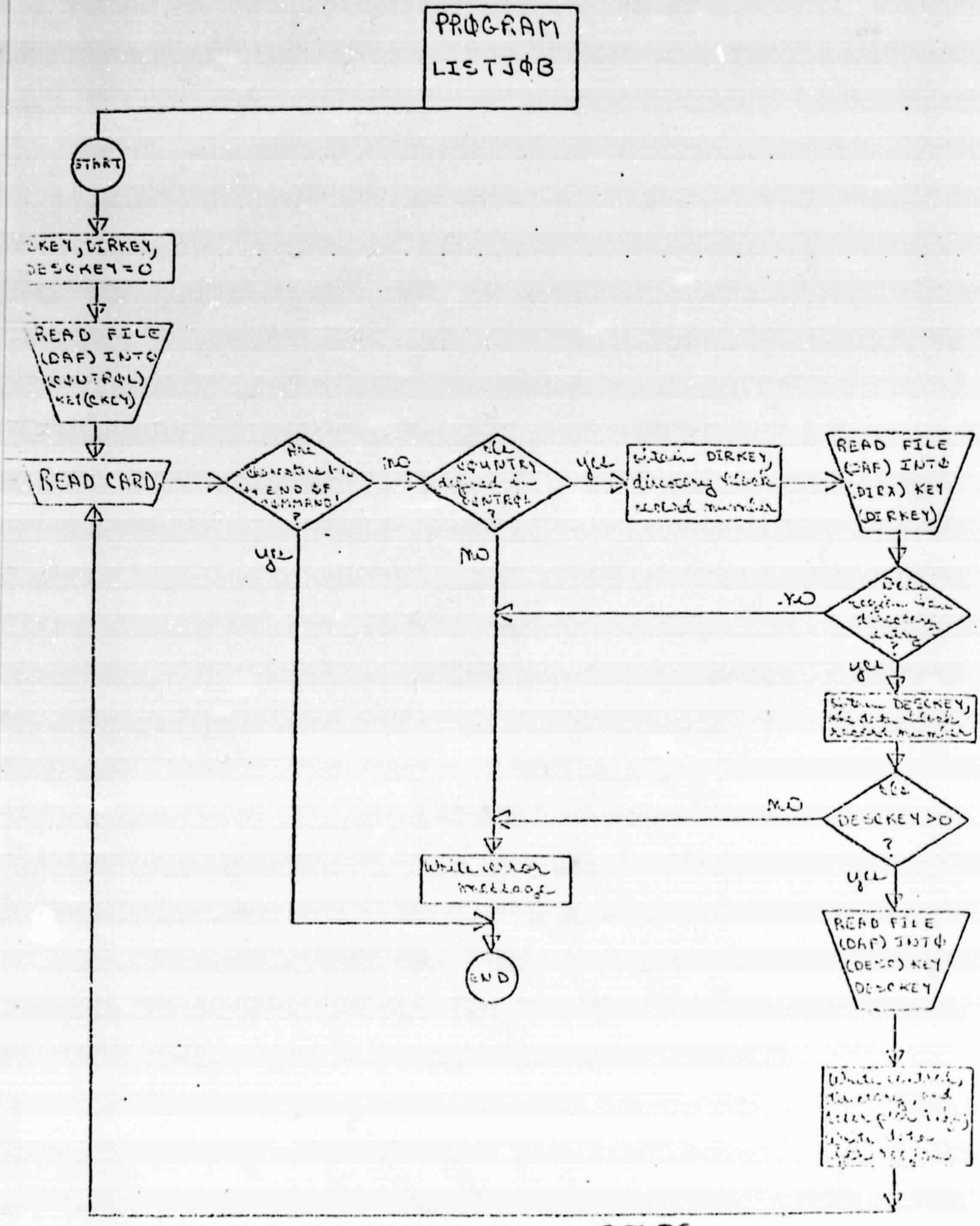
Listings of data by country.

3.3.15.5 Flow Chart

Next page.

3.3.15.6 Listing

Follows flow chart.


PROGRAM
LISTJOB
START
KEY, DIRKEY,
DESCKEY = 0
READ FILE
(DAF) INTO
(CONTROL)
KEY(CKEY)
READ CARD
END OF COMMAND?
NO
YES
YES
NO
NO
YES
NO
YES
END

```
RUN NO. 15      DATE  11/12/76    TIME  0910          LISTING OF MODULE LISTDATA
DESCRIPTION     LIST DATA BASE PGM

MASTER FILE     W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER 11/13/76
LAST DATE COPIED   NONE
LAST UPDATE        NONE

PASSWORD        DZKZ
PROGRAMMER      LEC
LANGUAGE        PLI
PROC PARAMETER  SNOJCL

LISTJOB: PROC OPTIONS(MAIN);
/*                                                                    */
/*                                                                    */
/*
   DCL 1 CONTROL,
         2 FILEID CHAR(8),
         2 NUMPASS FIXED BIN(15,0),
         2 PASS(8) CHAR(8),
         2 NUMLEV FIXED BIN(15,0),
         2 LEVNAME(8) CHAR(24),
         2 NUMCODE FIXED BIN(15,0),
         2 CODE(32),
           3 CODENUM FIXED BIN(15,0),
           3 UNITNUM FIXED BIN(15,0),
           3 BASE FIXED BIN(15,0),
           3 SCALE FIXED BIN(15,0),
           3 CODENAME CHAR(24),
           3 UNITNAME CHAR(24),
         2 NUMONE FIXED BIN(15,0),
         2 ONE(24),
           3 CODENUM FIXED BIN(15,0),
           3 NUMDIRS FIXED BIN(15,0),
           3 RECORD FIXED BIN(15,0),
           3 DISPLACE FIXED BIN(15,0),
           3 NAME CHAR(24),
         2 FILERECS FIXED BIN(15,0),
         2 REC(501),
           3 RECTYPE FIXED BIN(15,0),
           3 FREESPACE FIXED BIN(15,0),
           3 LOCATION FIXED BIN(15,0);
   DCL 1 DIRX,
         2 DIR (84),
           3 LEVNUM FIXED BIN(15,0),
           3 CODENUMB FIXED BIN(15,0),
           3 LAT FIXED BIN(15,0),
           3 LON FIXED BIN(15,0),
           3 UNNAME CHAR(24),
           3 PREC FIXED BIN(15,0),
           3 PDISP FIXED BIN(15,0),
           3 BREC FIXED BIN(15,0),
           3 BDISP FIXED BIN(15,0),
           3 CREC FIXED BIN(15,0),
           3 CDISP FIXED BIN(15,0),
           3 DREC FIXED BIN(15,0),
           3 DDISP FIXED BIN(15,0),
           3 LEVCODE FIXED BIN(31,0),
           3 MODEL(4),
             4 CROP FIXED BIN(15,0),
             4 XREC FIXED BIN(15,0),
             4 MDISP FIXED BIN(15,0),
         2 FILLER CHAR (56);
DCL 1 DESCDATA1,
      2 DESC1,
        3 ID FIXED BIN(31,0),
        3 XNO FIXED BIN(31,0),
        3 FILLER1 CHAR(4),
        3 ELEV FIXED BIN(15,0),
        3 TOTBLKS FIXED BIN(15,0),
        3 BLKSUSED FIXED BIN(15,0),
        3 BLKSIZE FIXED BIN(15,0),
        3 FSTREC FIXED BIN(15,0),
        3 FSTDISP FIXED BIN(15,0),
        3 LSTREC FIXED BIN(15,0),
        3 LSTDISP FIXED BIN(15,0),
        3 FILLER2 CHAR(18),
        3 NUMCODE FIXED BIN(15,0),
```


8990

```
        3 CODE(12),
          4 CODENUM FIXED BIN(15,0),
          4 NUMELEM FIXED BIN(15,0),
          4 ELEMSIZE FIXED BIN(15,0),
          4 NUMSCODE FIXED BIN(15,0),
          4 SUBCODE(8) FIXED BIN(15,0),
      2 DATA CHAR(4104);
DCL 1 DESCDATA2,
      2 DAT1 CHAR(3220),
      2 DESC2,
        3 ID FIXED BIN(31,0),
        3 WNO FIXED BIN(31,0),
        3 FILLER1 CHAR(4),
        3 ELEV FIXED BIN(15,0),
        3 TOTBLKS FIXED BIN(15,0),
        3 BLKSUSED FIXED BIN(15,0),
        3 BLKSIZE FIXED BIN(15,0),
        3 FSTREC FIXED BIN(15,0),
        3 FSTDISP FIXED BIN(15,0),
        3 LSTREC FIXED BIN(15,0),
        3 LSTDISP FIXED BIN(15,0),
        3 FILLER2 CHAR(18),
        3 NUMCODE FIXED BIN(15,0),
        3 CODE(12),
          4 CODENUM FIXED BIN(15,0),
          4 NUMELEM FIXED BIN(15,0),
          4 ELEMSIZE FIXED BIN(15,0),
          4 NUMSCODE FIXED BIN(15,0),
          4 SUBCODE(8) FIXED BIN(15,0),
      2 DAT2 CHAR(2884);
DCL 1 US,
      2 DESC CHAR(336),
      2 DATA(47),
        3 YP FIXED BIN(15,0),
        3 NXTYP FIXED BIN(15,0),
        3 NXTDISP FIXED BIN(15,0),
        3 FILLER1 FIXED BIN(15,0),
        3 TEMP(12) FIXED BIN(15,0),
        3 PRCP(12) FIXED BIN(15,0),
        3 DEGDY(12) FIXED BIN(15,0),
        3 HARV(4) FIXED BIN(31,0),
        3 PLANT(4) FIXED BIN(31,0),
        3 PROD(4) FIXED BIN(31,0),
      2 FILLER2 CHAR(88);
DCL 1 RUSSIA,
      2 USR(2),
        3 DESC CHAR(336),
        3 DATA(32),
          4 YP FIXED BIN(15,0),
          4 NXTYP FIXED BIN(15,0),
          4 NXTDISP FIXED BIN(15,0),
          4 FILLER1 FIXED BIN(15,0),
          4 TEMP(12) FIXED BIN(15,0),
          4 PRCP(12) FIXED BIN(15,0),
          4 HARV(4) FIXED BIN(31,0),
          4 PROD(4) FIXED BIN(31,0),
        3 FILL CHAR(48);
DCL 1 CANADA,
      2 DESC CHAR(336),
      2 DATA(47),
        3 YP FIXED BIN(15,0),
        3 NXTYP FIXED BIN(15,0),
        3 NXTDISP FIXED BIN(15,0),
        3 FILLER1 FIXED BIN(15,0),
        3 NXTP(12) FIXED BIN(15,0),
        3 NNTP(12) FIXED BIN(15,0),
        3 TEMP(12) FIXED BIN(15,0),
        3 PRCP(12) FIXED BIN(15,0),
        3 PLANT(3) FIXED BIN(31,0),
        3 PROD(3) FIXED BIN(31,0),
      2 FILLER2 CHAR(88);
DCL (CKEY,DIRKEY,DESCKEY,DESCDISP,COUNTRY,DIRNUM) FIXED BIN(15,0);
DCL (I,J,K,L,M,KK) FIXED BIN(15,0);
DCL CODES FIXED BIN(31,0);
DCL SYSIN FILE STREAM INPUT;
DCL SYSPRINT FILE STREAM OUTPUT;
DCL DAF FILE RECORD DIRECT KEYED ENV(REGIONAL(1));
DCL INSTR CHAR(80);
DCL SUBSTR BUILTIN;
ON ENDFILE(SYSIN) BEGIN;
  PUT SKIP FILE(SYSPRINT) EDIT
   ('***ERROR - END OF FILE SYSIN ENCOUNTERED***') (A);
  RCODE=-1;
  GOTO EOJ;
END;
RCODE=1;
CKEY=0;
READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
GETCARD: GET FILE(SYSIN) EDIT(INSTR) (COL(1),A(80));
```

```
   IF SUBSTR(INSTR,1,16)='**END OF COMMAND' THEN GOTO EOJ;
   GET STRING(INSTR) EDIT (CODES) (X(1),F(10,0));
   COUNTRY = FLOOR(CODES/10000000);
   CKEY,DIRKEY,DESCKEY,DIRNUM=0;
   I,J,K,L,M,DESCDISP,KK=0;
   DO I = 1 TO CONTROL.NUMONE;
     IF COUNTRY=CONTROL.ONE(I).CODE1NUM THEN DO;
       DIRKEY=CONTROL.ONE(I).RECNUM;
       DIRNUM=CONTROL.ONE(I).NUMDIRS;
       PUT PAGE FILE(SYSPRINT) EDIT (' THE COUNTRY '.CONTROL.ONE(I).NAME,
         'HAS '.CONTROL.ONE(I).NUMDIRS,' DIRECTORY ENTRIES ON RECORD '.
         DIRKEY) (A,A(24),A,F(4,0),A,F(4,0));
       PUT SKIP FILE(SYSPRINT) EDIT ('CODES  NAME','UNIT','BASE SCALE',
         ( CONTROL.CODE(J).CODENUM,CONTROL.CODE(J).CODENAME,
         CONTROL.CODE(J).UNITNAME,CONTROL.CODE(J).BASE,
         CONTROL.CODE(J).SCALE  DO J=1 TO CONTROL.NUMCODE)) (SKIP,X(5),A,
         X(21),A,X(17),A,SKIP,32(F(9,0),X(2),2 A(24),
         2 F(5,0),SKIP));
     END;
   END;
   IF DIRKEY>0 THEN DO;
     READ FILE(DAF) INTO(DIRX) KEY(DIRKEY);
     DO K = 1 TO DIRNUM;
       M=M+1;
       IF M=45 THEN DO;
         M=1;
         DIRKEY=DIRKEY+1;
         READ FILE(DAF) INTO(DIRX) KEY(DIRKEY);
       END;
       IF CODES=DIRX.DIR(M).LEVCODE THEN DO;
         DESCKEY=DIRX.DIR(M).DREC;
         DESCDISP=DIRX.DIR(M).DDISP;
         PUT SKIP(4) FILE(SYSPRINT) EDIT ('LEVEL','CODE','NAME','LATITUDE',
           'LONGITUDE','BASE U','#NOTHER','CHILD','DATA','MODELS',
           DIR(M).LEVNUM,DIR(M).CODENUMB,DIR(M).DIRNAME,DIR(M).LAT,
           DIR(M).LONG,DIR(M).DREC,DIR(M).DDISP,DIR(M).DREC,DIR(M).DDISP,
           DIR(M).DREC,DIR(M).DDISP,DIR(M).DREC,DIR(M).DDISP,
           ( DIR(M).MODEL(J),DREC,DIR(M).MODEL(J),DDISP  DO J=1 TO 4))
           (A,X(1),A,X(2),A,X(19),A,X(1),A,X(2),A,X(4),A,X(2),A,X(6),A,
           X(5),A,SKIP,F(3,0),F(5,0),X(4),A(24),F'----,0',X(3),F'----,0',
           8 F(5,0),X(1),4(F(4,0),F(5,0)));
       END;
     END;
     IF DESCKEY>0 THEN DO;
       PUT SKIP(4) FILE(SYSPRINT) EDIT ('DATA DESCRIPTOR IS ON RECORD',
         DESCKEY,'ID','A#0','FLEV','YR.ALLOC','YR.USED','BLKSIZE',
         'FSTREC','LSTREC','#CODE')
         (A,F(4,0),SKIP(2),X(3),A,X(6),A(A,X(1)),A,X(2),A,X(4),A,X(3),A);
       IF DESCDISP=1 THEN READ FILE(DAF) INTO(DESCDATA1) KEY(DESCKEY);
       IF DESCDISP=321 THEN DO;
         READ FILE(DAF) INTO(DESCDATA2) KEY(DESCKEY);
         DESCDATA1.DESC1=DESCDATA2.DESC2, BY NAME;
       END;
       PUT SKIP FILE(SYSPRINT) EDIT (DESC1.ID,DESC1.A#0,DESC1.FLEV,
         DESC1.TOTBLKS,DESC1.BLKSUSED,DESC1.BLKSIZE,DESC1.FSTREC,
         DESC1.FSTDISP,DESC1.LSTREC,DESC1.LSTDISP,DESC1.#CODE,
         '  DATA CODES   SUBCODE NUMBERS',(DESC1.CODE(J).CODENUM,
         (DESC1.CODE(J).SUBCODE(L) DO L=1 TO 4) DO J=1 TO 12))
         (F(11,0),2 F(5,0),2 F(7,0),F(9,0),X(1),4 F(5,0),F(6,0), SKIP(2),
         X(5),A, SKIP, 12(F(13,0),X(6),8 F(4,0),SKIP));
       PUT PAGE;
       IF COUNTRY=5 THEN DO;
         READ FILE(DAF) INTO(CANADA) KEY(DESCKEY);
         DO L = 1 TO DESC1.BLKSUSED;
         KK=KK+1;
           PUT SKIP(2) FILE(SYSPRINT) EDIT ('YEAR','NAT.YR.REC',
           'NAT.YR.DISP',CANADA.DATA(L).YR,CANADA.DATA(L).NXTYR,
           CANADA.DATA(L).NXTDISP,
           'CROP CODE          ',(DESC1.CODE(6).SUBCODE(J) DO J=1 TO 3),
           'PLANTED(HECTARES)  ',(CANADA.DATA(L).PLANT(J) DO J=1 TO 3),
           'PRODUCTION(QUINTALS)',(CANADA.DATA(L).PROD(J) DO J=1 TO 3),
           'JAN   FEB   MARCH  APRIL   MAY   JUNE   JULY   AUG   SEPT',
           '  OCT   NOV   DEC',
           'MAX TEMP(CENTIGRADE)',(CANADA.DATA(L).MXTP(J) DO J=1 TO 12),
           'MIN. TEMP(CENTIGRADE)',(CANADA.DATA(L).MNTP(J) DO J=1 TO 12),
           'MEAN TEMP(CENTIGRADE)',(CANADA.DATA(L).TEMP(J) DO J=1 TO 12),
           'PRECIP(MILLIMETERS)  ',(CANADA.DATA(L).PRCP(J) DO J=1 TO 12))
           (2(5,X(2),A, SKIP, F(4,0),X(5),F(1,0),X(3),F(4,0), SKIP(2),
           A(X(10),A,X(7),3 F(10,0), SKIP ),SKIP,X(34),A,A,SKIP,
           3(X(10),A,12 F'----,0',SKIP),X(10),A,12 F(7,0));
           IF KK=4 THEN DO;
             PUT PAGE FILE(SYSPRINT);
             KK=0;
           END;
```

```
        END;
      END;
      ELSE IF COUNTRY=8 THEN DO;
        READ FILE(DAF) INTO(RUSSIA) KEY(DESCKEY);
        IF DESC1ISP=1 THEN M=1;
        IF DESC1ISP=321 THEN M=2;
        DO L = 1 TO DESC1.BLKSUSED;
        KK=KK+1;
          PUT SKIP(2) FILE(SYSPRINT) EDIT ('YEAR'.'NXT.YR.REC'.
          'NXT.YR.DISP'.USR(M).DATA(L).YR.USR(M).DATA(L).NXTYR.
          USR(M).DATA(L).NXTDISP.
          'CROP CODE                '.(DESC1.CODE(8).SUBCODE(J) DO J=1 TO 4).
          'HARVESTED(HECTARES)  '.(USR(M).DATA(L).HARV(J) DO J=1 TO 4).
          'PRODUCTION(QUINTALS)'.(USR(M).DATA(L).PROD(J) DO J=1 TO 4).
          'JAN    FEB    MARCH   APRIL    MAY    JUNE    JULY     AUG    SEPT '.
          '    OCT     NOV     DEC'.
          'MEAN TEMP(CENTIGRADE)'.(USR(M).DATA(L).TEMP(J) DO J=1 TO 12).
          'PRECIP(MILLIMETERS)  '.(USR(M).DATA(L).PRCP(J) DO J=1 TO 12))
          (2(A,X(2)),A, SKIP, F(4,0),X(5),F(3,0),X(4),F(4,0), SKIP(2),
          3(X(10),A,X(7),A, F(10,0), SKIP), SKIP, X(34),A,A, SKIP,
          X(10),A,12 P'----9.9',SKIP, X(10),A,12 F(7,0));
          IF KK=4 THEN DO;
            PUT PAGE FILE(SYSPRINT);
            KK=0;
          END;
        END;
      END;
      ELSE IF COUNTRY=3 THEN DO;
        READ FILE(DAF) INTO(US) KEY(DESCKEY);
       DO L=1 TO DESC1.BLKSUSED;
        KK=KK+1;
          PUT SKIP(2) FILE(SYSPRINT) EDIT ('YEAR'.'NXT.YR.REC'.
          'NXT.YR.DISP'.US.DATA(L).YR.US.DATA(L).NXTYR.
          US.DATA(L).NXTDISP.
          'CROP CODE                '.(DESC1.CODE(7).SUBCODE(J) DO J=1 TO 4).
          'HARVESTED(HECTARES)  '.(US.DATA(L).HARV(J) DO J=1 TO 4).
          'PLANTED(HECTARES)    '.(US.DATA(L).PLANT(J) DO J=1 TO 4).
          'PRODUCTION(QUINTALS)'.(US.DATA(L).PROD(J) DO J=1 TO 4).
          'JAN    FEB    MARCH   APRIL    MAY    JUNE    JULY     AUG    SEPT '.
          '    OCT     NOV     DEC'.
          'MEAN TEMP(CENTIGRADE)'.(US.DATA(L).TEMP(J) DO J=1 TO 12).
          'PRECIP(MILLIMETERS)  '.(US.DATA(L).PRCP(J) DO J=1 TO 12).
          'DEGREE DAYS ABOVE    '.(US.DATA(L).DEGDY(J) DO J=1 TO 12))
          (2(A,X(2)),A, SKIP, F(4,0),X(5),F(3,0),X(4),F(4,0), SKIP(2),
          4(X(10),A,X(7),A F(10,0), SKIP), SKIP, X(34),A,A, SKIP,
          X(10),A,12 P'----9.9',SKIP,X(10),A,12 F(7,0),SKIP,
          X(10),A,12 P'----9.9');
          IF KK=4 THEN DO;
            PUT PAGE FILE(SYSPRINT);
            KK=0;
          END;
        END;
       END;
       ELSE PUT SKIP(2) FILE(SYSPRINT) EDIT('DESIRED COUNTRY') (A);
    END;
    ELSE PUT SKIP(2) FILE(SYSPRINT) EDIT ('COUNTRY HAS NO DATA') (A);
  END;
  ELSE PUT SKIP(2) FILE(SYSPRINT)
    EDIT('COUNTRY HAS NO DIRECTORY ENTRY') (A);
  GOTO GETCARD;
  EOJ: END LISTJOB;
```

4. OPERATION

This section describes the operation of each of the monthly yield data base support programs.

4.1 OPERATING INSTRUCTIONS

There are four types of programs run in maintaining and using the YES Yield Monthly Data Base: data base initialization, data base definition, data base load and update and listing.

4.1.1 DATA BASE INITIALIZATION

The file initialization program is the first program to be run in setting up the data base. This program defines the first record of the file to be the control block and all other records as blank. It also sets the variables in the control block to some dummy values which will be changed in subsequent programs to accommodate the actual situation. One variable, the number of records contained in the file excluding the control block, is dependent on the user's facilities and must be filled into the program before it is run.

4.1.2 DATA BASE DEFINITION

Definition of the data base involves establishing the control block, defining the directories, and entering the data definitions. An example run setup is given in appendix C.

4.1.2.1 Definition of the Control Block

Definition of the control block is the second step in the creation of the data base.

1. Not all sections and subsections must be defined by the user. The file initialization program sets all variables to standard values and some of these values should be changed only when they are automatically modified during execution of other programs.

These include the number of level-one entries, the information about level-one entries, the number of records on the file, and the information about each of the records on the file; these are sections 8, 9, 10, and 11, respectively. All other sections should be defined.

2. Information is read in on cards with only one section or only one subsection of a section on a card.

3. Names are punched left-justified, or starting in the leftmost column of the field, and numbers are punched right-justified.

4. The section number must be punched in columns 6 and 7, and the subsection number in columns 9 and 10. Zero is used if the section has no subsection.

5. Since each section contains different types of information, the formats in which they are entered must also change.

   a. For sections 2, 4, 6, 8, and 10, the appropriate number is punched in the field of columns 12 to 15.

   b. For section 1, the file identification name is punched in the field of columns 12 to 19.

   c. For all subsections of section 3, the password is punched in the field of columns 12 to 19. Note that once sections 2 and 3 are defined, subsequent programs accessing the file will require a password card.

   d. For all subsections of section 5, the level name is punched in the field of columns 12 to 35.

   e. For all subsections of section 7, the code number, unit number, base, scale, code name, and unit name should be punched in the field of columns 12 to 15, 17 to 20, 22 to 25, 27 to 30, 32 to 55, and 57 to 80, respectively.

   f. For all subsections of section 9, the code number, number of directories, record number, displacement, and name

should be punched in the field of columns 12 to 15, 17 to 20, 22 to 25, 27 to 30, and 32 to 55, respectively.

g. For all subsections of section 11, the record type and amount and location of free space should be punched in the field of columns 12 to 15, 17 to 20, and 22 to 25, respectively.

Updating the control block is done in two ways, manually by the user or automatically with the other programs.

1. The manual update of the control block is done with the same program that was used for defining it. Consequently, the same formats for each of the sections and/or subsections are followed. Any section and/or subsection can be changed using the program but only sections 1 through 7 or section 10 should ever need to be changed. For the subsections of sections 7, 9, and 11, all the variables must be punched on the card even if some values remain the same; if a variable's field is empty, it will be coded as blank on the file.

2. The automatic update of the control block is done by the other programs which add information to the file. The sections 8 and 9 are changed when the directory block for a new level-one region is defined. Whenever a block on the file is read into for the first time, section 11 is changed to show which type of information was read in; also whenever information is added in a block, section 11 is changed to show the amount of free space remaining.

4.1.2.2 Defining the Directories

Definition of the directory entries in the directory block, or blocks, is the third step in the creation of the data base.

1. Information for each directory entry is read in on one card.

2. Names are punched left-justified and numbers are punched right-justified in their appropriate field of columns.

3. Sections 1 through 5 and section 14 are punched in columns 3 to 4, 5 to 8, 10 to 14, 15 to 19, 21 to 44, and 71 to 80, respectively. These are the level number, code number, latitude, longitude, entry name and the unique ten-digit code.

4. Sections 12 and 13 are defined during execution of the program which defines the data descriptor entries, and section 15 is defined during execution of the program which defines the model definition blocks; no user definition is required.

5. Sections 6 through 11 are defined with the define directory program, but some user input is necessary. In the field of columns 45 to 48, the position in the input card deck of the entry's parent is coded. For example, if Colorado, a level-three region, is the third directory entry card, then the entries for the level-four regions in Colorado would have a 3 coded in column 48. Level-one regions would have a negative one coded since they have no parent. In the fields of columns 49 to 52 and 53 to 56 are coded the positions in the card deck of the directory entries corresponding to the entry's brother and child. Negative ones are coded if there is no brother or child.

6. Only directory entries for one level-one region and the higher levels within it can be defined during one execution of the define directory program. The program will be terminated if a second level-one card is encountered.

7. The first input card must be the level-one region's directory entry. If the level of the first card is not one, then the program will be terminated. The remaining cards can be in any order; however, calculations of the parent, brother, and child positions would be facilitated if the entries were kept in sequence.

8. If the define directory program is run twice with the same input cards, then there will be two directory blocks for the same country, and the country will be listed twice in the control block information.

Directory entries are automatically updated by the other programs which add information to the file. Sections 12 and 13 are changed when the data descriptor entries are added to the file. The subsections of section 15 are changed when the model definition blocks are added to the file.

4.1.2.3 Defining the Data Descriptors

Definition of the data descriptor entries must be done before the data can be placed on the file and after the directory entries have been defined.

1. Information for each data descriptor entry is read in on a set of cards, the number of cards dependent on the number of variable codes required for the data.
2. Numbers are punched right-justified in their appropriate field of columns.
3. Sections 1, 2, 5, 6, 8, and 14 are punched in columns 2 to 11, 13 to 17, 19 to 22, 24 to 25, 27 to 30, and 32 to 33, respectively. These are the identification number, WMO number, elevation, total number of years for which data could be defined, length in bytes for storage of one year's data, and the number of codes.
4. The information for each code in section 15 is punched on a separate card. The number of code cards must be equal to the number of codes specified on the first card. The code number, number of elements, element size, and number of subcodes are punched in the field of columns 2 to 4, 6 to 8, 10 to 11, and 13, respectively. The one to eight subcode numbers are punched

in the fields of columns 15 to 17, 19 to 21, 23 to 25, 27 to 29, 31 to 33, 35 to 37, 39 to 41, and 43 to 45, as needed.

5. Sections 3 and 4 are defined during execution of the define descriptor program by copying the information from the region's directory entry. Sections 7 and 9 to 12 are defined during execution of the program which defines the data onto the file. No user definition is required.

6. The entire set of cards is repeated for each data descriptor entry being defined.

Updating the data descriptor entries is done two ways, manually by the user and automatically with the define data programs.

1. The manual update of the descriptor entries is done with the same program that was used for defining the entries. In order to update a particular descriptor entry which is already defined, the entire set of cards used to define that entry is input again with appropriate corrections made. The pointers to the data, sections 9 to 12, are not changed when the define descriptor program is used for update. To add more data descriptor entries to the file, the same format is used to construct the set of cards for each entry and the define descriptor program used again.

2. The automatic update of the data descriptor entries is done during execution of programs which define or update data on the file. The sections involved are 7 and 9 to 12.

4.1.3 ENTERING AND UPDATING DATA

Initial load of data may be done either by the updating program UPDDATA, or the individual country loaders AUSARG, USSR, CANADA, and USA.

4.1.3.1 Entering Data With the Individual Country Loaders

Before the data can be placed on the file the control block, directory entries and data descriptor entries must all be defined.

1. Data for each variable within a certain year and region are entered on separate cards. The cards are grouped by year and sorted chronologically within each region before execution of a define data program.

2. Numbers are punched right-justified in their appropriate field of columns.

3. There are two different formats for entering data, one for meteorological data and one for yield data. Both formats require the year, variable code, and identification number to be punched in the field of columns 4 to 7, 8 to 10, and 71 to 80, respectively.

   a. For a meteorological variable, the data for each of the 12 months are punched in the field of columns 11 to 15, 16 to 20, 21 to 25, 26 to 30, 31 to 35, 36 to 40, 41 to 45, 46 to 50, 51 to 55, 56 to 60, 61 to 65, and 66 to 70. If any of the 12 fields is blank, the value of the variable for that month will be coded as -9999 on the file to indicate a missing value.

   b. For a yield variable, the data for each crop are punched in the field of columns 11 to 20, 21 to 30, 31 to 40, and 41 to 50, as needed. If more than one crop is reported, then it is assumed that the data are organized in ascending order according to crop code. For example, spring wheat with code 201 is punched in the field 11 to 20 and winter wheat with code 202 in the field 21 to 30. If there is only one crop, the value of the yield variable is punched in the field 11 to 20. Extra fields should be left blank.

4.1.3.2 Updating Data

Data are updated by use of the update data program UPDDATA. Update includes changing data for years which already exist on the file and adding data for new years. It does not include adding data for regions which have no data descriptor entry; the data descriptor entry must be defined first.

1. The same formats used for defining data of the meteorological and yield variables are used for updating those data.

2. Cards can be entered in any order, although sorting the cards by year for each region identification number makes the program more efficient.

3. In the case of meteorological data, values of the variable for any month which are left blank will be assigned values of -9999. Therefore, if one month of a year's precipitation data needs to be changed, the values for all months should be coded. In the case of yield data, the same procedure will hold for the values of the variables for the different crops which are missing.

4. When a new year is added to the file, it is not necessary that all variables be defined; one variable card for a year not previously defined is sufficient to initialize space and change all appropriate pointers for the new year. However, values of the undefined variables will be zero rather than the value -9999 which usually denotes a missing value. The -9999 can be assigned for all values of a variable by entering a card for that variable with blanks for all months or crop information.

5. When an old year is updated, only the variable or variables which need changes need input cards. The other variables remain unchanged.

6. The program assumes that enough free space exists in the data block for extra years if they need to be defined. It does not start a new record as a second data block for the region.

7. The program assumes that the variable being updated or defined is one which is already defined in the region's data descriptor entry. It cannot define new variables until their code number, position, and length are put in the data descriptor.

4.1.4 LISTING PROGRAMS

Three listing programs are provided: YESLS02 to list the control block, YESLS04 to list directories, and LISTJOB to list data.

4.1.4.1 Listing the Control Block

To list information in the control block, the program YESLS02 is used. The only input required is a card with '++END OF COMMAND' punched in columns 1 to 16.

4.1.4.2 Listing the Directory Blocks

To list directory information, the program YESLS04 is used. The ten-digit identification code for the appropriate region should be punched in columns 2 to 11. The program will then list the directory entry for that region and all smaller regions within that region. For example, if all the Canadian directory entries are needed, the code 0500000000 is used; if only the Alberta regions are needed, then the code 0502030000 is used. Any number of input code cards can be used, and then all followed by an '++END OF COMMAND' card.

4.1.4.3 Listing the Data Descriptor and Data Blocks

To list descriptor information and data, the program LISTJOB is used. The ten-digit identification code for the appropriate region should be punched in columns 2 to 11. Some control and directory information will be printed as well as data for all available years for that region. Any number of input code cards can be used, and then all followed by an '++END OF COMMAND' card.

APPENDIX A

STRUCTURES

Appendix A: Structures

Control Block
This is the first record on the file and is 6440 bytes long.

```
DCL 1 CONTROL
      2 FILEID          CHAR(8),
      2 NUMPASS         FIXED BIN(15,0),
      2 PASS(8)         CHAR(8),
      2 NUMLEV          FIXED BIN(15,0),
      2 LEVNAME(8)      CHAR(24),
      2 NUMCODE         FIXED BIN(15,0),
      2 CODE(32),
        3 CODENUM       FIXED BIN(15,0),
        3 UNITNUM       FIXED BIN(15,0),
        3 BASE          FIXED BIN(15,0),
        3 SCALE         FIXED BIN(15,0),
        3 CODENAME      CHAR(24),
        3 UNITNAME      CHAR(24),
      2 NUMONE          FIXED BIN(15,0),
      2 ONE(24),
        3 CODE1NUM      FIXED BIN(15,0),
        3 NUMDIRS       FIXED BIN(15,0),
        3 RECNUM        FIXED BIN(15,0),
        3 DISPLACE      FIXED BIN(15,0),
        3 NAME          CHAR(24),
      2 FILERECS        FIXED BIN(15,0),
      2 REC(601),
        3 RECTYPE       FIXED BIN(15,0),
        3 FRESPACE      FIXED BIN(15,0),
        3 LOCATION      FIXED BIN(15,0);
```

Directory Entry
A maximum of 84 directory entries can be placed in a directory block; each entry is 76 bytes long.

```
DCL 1 DIR,
      2 LEVNUM       FIXED BIN(15,0),
      2 CODENUMB     FIXED BIN(15,0),
      2 LAT          FIXED BIN(15,0),
      2 LON          FIXED BIN(15,0),
      2 DIRNAME      FIXED BIN(15,0),
      2 PREC         FIXED BIN(15,0),
      2 PDISP        FIXED BIN(15,0),
      2 BREC         FIXED BIN(15,0),
      2 BDISP        FIXED BIN(15,0),
      2 CREC         FIXED BIN(15,0),
      2 CDISP        FIXED BIN(15,0),
      2 DREC         FIXED BIN(15 0),
      2 DDISP        FIXED BIN(15,0),
      2 LEVCODE      FIXED BIN(31,0),
      2 MODEL(4),
        3 CROP       FIXED BIN(15,0),
        3 MREC       FIXED BIN(15,0),
        3 MDISP      FIXED BIN(15,0);
```

Data Descriptor Entry
This precedes the data for each region in the data blocks; it is 336 bytes long.

```
DCL 1 DESC,
      2 ID              FIXED BIN(31,0),
      2 WMO             FIXED BIN(31,0),
      2 LATI            FIXED BIN(15,0),
      2 LONG            FIXED BIN(15,0),
      2 ELEV            FIXED BIN(15,0),
      2 TOTBLKS         FIXED BIN(15,0),
      2 NUMBYRS         FIXED BIN(15.0),
      2 BLKSIZE         FIXED BIN(15,0),
      2 FSTRECNO        FIXED BIN(15,0),
      2 FSTDISP         FIXED BIN(15,0),
      2 LSTRECNO        FIXED BIN(15,0),
      2 LSTDISP         FIXED BIN(15,0),
      2 RESERVED        CHAR(18),
      2 NUMBCODE        FIXED BIN(15,0),
      2 DCODE(12),
        3 CODENUMB      FIXED BIN(15,0),
        3 NUMSELEM      FIXED BIN(15,0),
        3 ELEMSIZE      FIXED BIN(15,0),
        3 NUMSCODE      FIXED BIN(15,0),
        3 SUBCODE(8)    FIXED BIN(15,0);
```

Australia Data Year Entry
There is a maximum of 47 years following the data descriptor entry in a data block for each Australian region; each year entry is 128 bytes long.

```
DCL 1 AUSTRALIA,
      2 YEAR             FIXED BIN(15,0),
      2 NXTYRREC         FIXED BIN(15 0),
      2 NXTYRDISP        FIXED BIN(15,0),
      2 FILLER(17)       FIXED BIN(15,0),
      2 MEANTEMP(12)     FIXED BIN(15,0),
      2 PRECIP(12)       FIXED BIN(15,0),
      2 Z(12)            FIXED BIN(15,0),
      2 PRODUCTION(2)    FIXED BIN(31,0),
      2 HARVESTED(2)     FIXED BIN(31,0);
```

Canada Data Year Entry
There is a maximum of 47 years following the data descriptor entry in a data block for each Canadian region; each year entry is 128 bytes long.

```
DCL 1 CANADA,
      2 YEAR             FIXED BIN(15,0),
      2 NXTYRREC         FIXED BIN(15,0),
      2 NXTRYDISP        FIXED BIN(15,0),
      2 FILLER           FIXED BIN(15,0),
      2 MAXTEMP(12)      FIXED BIN(15,0),
      2 MINTEMP(12)      FIXED BIN(15,0),
      2 MEANTEMP(12)     FIXED BIN(15,0),
      2 PRECIP(12)       FIXED BIN(15,0),
      2 PLANTED(3)       FIXED BIN(31,0),
      2 PRODUCTION(3)    FIXED BIN(31,0);
```

U.S.S.R. Data Year Entry
There is a maximum of 22 years following the data descriptor entry for each Russian region in a data block; the data for two regions can be placed in each data block. Each year entry is 88 bytes long.

```
DCL 1 USSR,
      2 YEAR            FIXED BIN(15,0),
      2 NXTYRREC        FIXED BIN(15,0),
      2 NXTYRDISP       FIXED BIN(15,0),
      2 FILLER          FIXED BIN(15,0),
      2 MEANTEMP(12)    FIXED BIN(15,0),
      2 PRECIP(12)      FIXED BIN(15,0),
      2 HARVESTED(4)    FIXED BIN(31,0),
      2 PRODUCTION(4)   FIXED BIN(31,0);
```

United States Data Year Entry
There is a maximum of 47 years following the data descriptor entry in a data block for each United States region; each year entry is 128 bytes long.

```
DCL 1 US,
      2 YEAR            FIXED BIN(15,0),
      2 NXTYRREC        FIXED BIN(15,0),
      2 NXTYRDISP       FIXED BIN(15,0),
      2 FILLER          FIXED BIN(15,0),
      2 MEANTEMP(12)    FIXED BIN(15,0),
      2 PRECIP(12)      FIXED BIN(15,0),
      2 DEGREEDAY(12)   FIXED BIN(15,0),
      2 HARVESTED(4)    FIXED BIN(31,0),
      2 PLANTED(4)      FIXED BIN(31,0),
      2 PRODUCTION(4)   FIXED BIN(31,0);
```

APPENDIX B

VARIABLE CODES

Appendix B: Variables Codes

Meteorological Variables	
Precipitation	5
Maximum Temperature	15
Minimum Temperature	25
Mean Temperature	35
Degree Days Above	40
Degree Days Below	50
Palmer Drought Z-Index	45
Yield Variables	
Harvested	101
Planted	102
Production	103
Harvested Yield	104
Planted Yield	105
Crops	
Spring Wheat	201
Winter Wheat	202
Rice	206
Corn	211
Soybeans	216
Sorghum	221
Flax	226
Unit of Measurement	
Inches	102
Bushels	128
Acres	136
Degrees Fahrenheit	141
Bushels/Acre	151
Millimeters	201
Quintals	228
Hectares	236
Degrees Centigrade	241
Quintals/Hectare	251
Monthly	5
Others	
Hourly	1
3-hourly	3
6-hourly	6
Daily	11
Weekly	16
Monthly	26
Year	61
Pointer	90
Record Pointer	91
Displacement Pointer	92
Filler or Reserved Space	99

APPENDIX C

SAMPLE INPUT TO YESM001

```
RUN NO. 7        DATE  10/28/76     TIME  1106             LISTING OF MODULE DBDATA

DESCRIPTION           DEFINE CNTRL&DIR&DESC/CANARUS

MASTER FILE           W.FDS.CCEA.LEC.LIBR
ADDED TO MASTER       10/28/76
LAST DATE COPIED        NONE
LAST UPDATE             NONE

PASSWORD              VDCC
PROGRAMMER            LFC
LANGUAGE              DAT
PROC PARAMETER        $NOJCL

  //BCOOK JOB (*DD1000REIHEA   '.'COLUM').COOK.REGION=160K.TIME=1
  //   EXEC PGM=YESM001
  //STEPLIB DD DSN=W.FDS.CCEA.LEC.LOAD.DISP=SHR
  //SYSPRINT DD SYSOUT=A
  //DAF DD DSN=W.FDS.CCEA.MET.CLIMAT.DISP=SHR.DCB=BUFNO=1
  //SYSIN DD *
  ++COMMAND=DEFINE    ++OPERAND=CONTROL
        2  0    2
        3  1 COOK
        3  2 SLEMONS
        6  0    16
        7  1     5   201     2     0 PRECIPITATION                  MILLIMETERS
        7  2    15   241     2    -1 MAXIMUM TEMPERATURE            DEGREES CENTIGRADE
        7  3    25   241     2    -1 MINIMUM TEMPERATURE            DEGREES CENTIGRADE
        7  4    35   241     2    -1 MEAN TEMPERATURE               DEGREES CENTIGRADE
        7  5   101   235     4     0 HARVESTED                      HECTARES
        7  6   102   236     4     0 PLANTED                        HECTARES
        7  7   103   228     4     0 PRODUCTION                     QUINTALS
        7  8    50     5     2     0 DEGREE DAYS ABOVE              MONTHLY
        7  9   201     0     3    12 SPRING WHEAT
        7 10   202     0     3    12 WINTER WHEAT
        7 11    26     0     3     7 MONTHLY
        7 12    61     0     2     0 YEAR
        7 13    90     0     2     0 POINTER
        7 14    91     0     2     0 RECORD POINTER
        7 15    92     0     2     0 DISPLACEMENT POINTER
        7 16    94     0     2     0 FILLER
  ++END OF COMMAND
  ++COMMAND=DEFINE      ++OPERAND=DIRECTORY
     1    5     550 1050 CANADA                    -1  -1   2        500000000
     2    2     550 1050 PRAIRIE PROVINCES          1  -1   3        502000000
     3    3     556 1140 ALBERTA                    2   7   4        502030000
     4    1     515 1128 SOUTHERN                   3   5  -1        502030100
     4    2     555 1113 CENTRAL                    3   6  -1        502030200
     4    3     570 1171 NORTHERN                   3  -1  -1        502030300
     3    4     545 1060 SASKATCHEWAN               2  17   8        502040000
     4    1     496 1025 SOUTHEASTERN               7   9  -1        502040100
     4    2     497 1045 REGINA-WEYBURN             7  10  -1        502040200
     4    3     498 1075 SOUTH CENTRAL              7  11  -1        502040300
     4    4     498 1095 SOUTHWESTERN               7  12  -1        502040400
     4    5     513 1030 EAST CENTRAL               7  13  -1        502040500
     4    6     513 1030 CENTRAL                    7  14  -1        502040600
     4    7     515 1089 WEST CENTRAL               7  15  -1        502040700
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
 4  8  327 1035 NORTHEASTERN                    7  15  -1   5020408000  0004400
 4  9  565 1060 NORTHWESTERN                    1  -1  -1   5020409000  0004500
 5  5  550  970 MANITOBA                        2  -1  14   5020506000  0004600
 4  3  500  960 SOUTHEASTERN                   17  19  -1   5020501000  0004700
 4  4  495  975 SOUTH CENTRAL                  17  20  -1   5020502000  0004800
 4  4  495  992 SOUTHWESTERN                   17  21  -1   5020503000  0004900
 4  4  510  992 NORTHERN                       17  -1  -1   5020504000  0005000
++END OF COMMAND
++COMMAND=DEFINE    ++OPERAND=DESCRIPTOR                                 0005100
 5020301000 0 91 92                                                      0005200
 5020302000 0 91 92                                                      0005300
 5020303000 0 91 92                                                      0005400
 5020304000 0 91 92                                                      0005500
 5020305000 0 91 92                                                      0005600
 5020306000 0 91 92                                                      0005700
 5020307000 0 91 92                                                      0005800
 5020308000 0 91 92                                                      0005900
 5020309000 0 91 92                                                      0006000
 5020310000 0 91 92                                                      0006100
 5020311000 0 91 92                                                      0006200
 5020312000 0 91 92                                                      0006300
 5020313000 0 91 92                                                      0006400
 5020314000 0 91 92                                                      0006500
 5020315000 0 91 92                                                      0006600
 5020316000 0 91 92                                                      0006700
 5020317000 0 91 92                                                      0006800
 5020318000 0 91 92                                                      0006900
 5020319000 0 91 92                                                      0007000
 5020320000 0 91 92                                                      0007100
 5020321000 0 91 92                                                      0007200
 5020322000 0 91 92                                                      0007300
 5020323000 0 91 92                                                      0007400
 5020324000 0 91 92                                                      0007500
 5020325000 0 91 92                                                      0007600
 5020326000 0 91 92                                                      0007700
 5020327000 0 91 92                                                      0007800
 5020328000 0 91 92                                                      0007900
 5020329000 0 91 92                                                      0008000
 5020330000 0 91 92                                                      0008100
 5020331000 0 91 92                                                      0008200
 5020332000 0 91 92                                                      0008300
 5020333000 0 91 92                                                      0008400
 5020334000 0 91 92                                                      0008500
 5020335000 0 91 92                                                      0008600
 5020336000 0 91 92                                                      0008700
 5020337000 0 91 92                                                      0008800
 5020338000 0 91 92                                                      0008900
 5020339000 0 91 92                                                      0009000
 5020340000 0 91 92                                                      0009100
 5020341000 0 91 92                                                      0009200
 5020342000 0 91 92                                                      0009300
 5020343000 0 91 92                                                      0009400
 5020344000 0 91 92                                                      0009500
 5020345000 0 91 92                                                      0009600
 5020346000 0 91 92                                                      0009700
 5020347000 0 91 92                                                      0009800
 5020348000 0 91 92                                                      0009900
 5020349000 0 91 92                                                      0010000
 5020350000 0 91 92                                                      0010100
 5020351000 0 91 92                                                      0010200
 5020352000 0 91 92                                                      0010300
 5020353000 0 91 92                                                      0010400
 5020354000 0 91 92                                                      0010500
 5020355000 0 91 92                                                      0010600
 5020356000 0 91 92                                                      0010700
 5020357000 0 91 92                                                      0010800
 5020358000 0 91 92                                                      0010900
 5020359000 0 91 92                                                      0011000
 5020360000 0 91 92                                                      0011100
 5020361000 0 91 92                                                      0011200
 5020362000 0 91 92                                                      0011300
 5020363000 0 91 92                                                      0011400
 5020364000 0 91 92                                                      0011500
 5020365000 0 91 92                                                      0011600
 5020366000 0 91 92                                                      0011700
 5020367000 0 91 92                                                      0011800
 5020368000 0 91 92                                                      0011900
 5020369000 0 91 92                                                      0012000
 5020370000 0 91 92                                                      0012100
 5020371000 0 91 92                                                      0012200
 5020372000 0 91 92                                                      0012300
 5020373000 0 91 92                                                      0012400
 5020374000 0 91 92                                                      0012500
 5020375000 0 91 92                                                      0012600
 5020376000 0 91 92                                                      0012700
 5020377000 0 91 92                                                      0012800
 5020378000 0 91 92                                                      0012900
 5020379000 0 91 92                                                      0013000
 5020380000 0 91 92                                                      0013100
 5020381000 0 91 92                                                      0013200
 5020382000 0 91 92                                                      0013300
 5020383000 0 91 92                                                      0013400
 5020384000 0 91 92                                                      0013500
 5020385000 0 91 92                                                      0013600
 5020386000 0 91 92                                                      0013700
 5020387000 0 91 92                                                      0013800
 5020388000 0 91 92                                                      0013900
 5020389000 0 91 92                                                      0014000
 5020390000 0 91 92                                                      0014100
 5020391000 0 91 92                                                      0014200
 5020392000 0 91 92                                                      0014300
 5020393000 0 91 92                                                      0014400
 5020394000 0 91 92                                                      0014500
 5020395000 0 91 92                                                      0014600
 5020396000 0 91 92                                                      0014700
 5020397000 0 91 92                                                      0014800
 5020398000 0 91 92                                                      0014900
 5020399000 0 91 92                                                      0015000
 5020400000 0 91 92                                                      0015100
 5020401000 0 91 92                                                      0015200
 5020402000 0 91 92                                                      0015300
 5020403000 0 91 92                                                      0015400
 5020404000 0 91 92                                                      0015500
 5020405000 0 91 92                                                      0015600
 5020406000 0 91 92                                                      0015700
 5020407000 0 91 92                                                      0015800
 5020408000 0 91 92                                                      0015900
 5020409000 0 91 92                                                      0016000
 5020410000 0 91 92                                                      0016100
 5020411000 0 91 92                                                      0016200
 5020412000 0 91 92                                                      0016300
 5020413000 0 91 92                                                      0016400
 5020414000 0 91 92                                                      0016500
 5020415000 0 91 92                                                      0016600
 5020416000 0 91 92                                                      0016700
 5020417000 0 91 92                                                      0016800
 5020418000 0 91 92                                                      0016900
 5020419000 0 91 92                                                      0017000
 5020420000 0 91 92                                                      0017100
 5020421000 0 91 92                                                      0017200
 5020422000 0 91 92                                                      0017300
 5020423000 0 91 92                                                      0017400
 5020424000 0 91 92                                                      0017500
 5020425000 0 91 92                                                      0017600
 5020426000 0 91 92                                                      0017700
 5020427000 0 91 92                                                      0017800
 5020428000 0 91 92                                                      0017900
 5020429000 0 91 92                                                      0018000
 5020430000 0 91 92                                                      0018100
 5020431000 0 91 92                                                      0018200
 5020432000 0 91 92                                                      0018300
 5020433000 0 91 92                                                      0018400
 5020434000 0 91 92                                                      0018500
 5020435000 0 91 92                                                      0018600
 5020436000 0 91 92                                                      0018700
 5020437000 0 91 92                                                      0018800
 5020438000 0 91 92                                                      0018900
 5020439000 0 91 92                                                      0019000
 5020440000 0 91 92                                                      0019100
 5020441000 0 91 92                                                      0019200
 5020442000 0 91 92                                                      0019300
 5020443000 0 91 92                                                      0019400
 5020444000 0 91 92                                                      0019500
 5020445000 0 91 92                                                      0019600
 5020446000 0 91 92                                                      0019700
 5020447000 0 91 92                                                      0019800
 5020448000 0 91 92                                                      0019900
 5020449000 0 91 92                                                      0020000
 5020450000 0 91 92                                                      0020100
 5020451000 0 91 92                                                      0020200
 5020452000 0 91 92                                                      0020300
 5020453000 0 91 92                                                      0020400
 5020454000 0 91 92                                                      0020500
 5020455000 0 91 92                                                      0020600
 5020456000 0 91 92                                                      0020700
 5020457000 0 91 92                                                      0020800
 5020458000 0 91 92                                                      0020900
 5020459000 0 91 92                                                      0021000
 5020460000 0 91 92                                                      0021100
 5020461000 0 91 92                                                      0021200
 5020462000 0 91 92                                                      0021300
 5020463000 0 91 92                                                      0021400
 5020464000 0 91 92                                                      0021500
 5020465000 0 91 92                                                      0021600
 5020466000 0 91 92                                                      0021700
 5020467000 0 91 92                                                      0021800
 5020468000 0 91 92                                                      0021900
 5020469000 0 91 92                                                      0022000
 5020470000 0 91 92                                                      0022100
 5020471000 0 91 92                                                      0022200
 5020472000 0 91 92                                                      0022300
 5020473000 0 91 92                                                      0022400
 5020474000 0 91 92                                                      0022500
 5020475000 0 91 92                                                      0022600
 5020476000 0 91 92                                                      0022700
 5020477000 0 91 92                                                      0022800
 5020478000 0 91 92                                                      0022900
 5020479000 0 91 92                                                      0023000
 5020480000 0 91 92                                                      0023100
 5020481000 0 91 92                                                      0023200
 5020482000 0 91 92                                                      0023300
 5020483000 0 91 92                                                      0023400
 5020484000 0 91 92                                                      0023500
 5020485000 0 91 92                                                      0023600
 5020486000 0 91 92                                                      0023700
 5020487000 0 91 92                                                      0023800
 5020488000 0 91 92                                                      0023900
 5020489000 0 91 92                                                      0024000
 5020490000 0 91 92                                                      0024100
 5020491000 0 91 92                                                      0024200
 5020492000 0 91 92                                                      0024300
 5020493000 0 91 92                                                      0024400
 5020494000 0 91 92                                                      0024500
 5020495000 0 91 92                                                      0024600
 5020496000 0 91 92                                                      0024700
 5020497000 0 91 92                                                      0024800
 5020498000 0 91 92                                                      0024900
 5020499000 0 91 92                                                      0025000
 5020500000 0 91 92                                                      0025100
 5020501000 0 91 92                                                      0025200
 5020502000 0 91 92                                                      0025300
 5020503000 0 91 92                                                      0025400
 5020504000 0 91 92                                                      0025500
 5020505000 0 91 92                                                      0025600
 5020506000 0 91 92                                                      0025700
 5020507000 0 91 92                                                      0025800
 5020508000 0 91 92                                                      0025900
 5020509000 0 91 92                                                      0026000
 5020510000 0 91 92                                                      0026100
 5020511000 0 91 92                                                      0026200
 5020512000 0 91 92                                                      0026300
 5020513000 0 91 92                                                      0026400
 5020514000 0 91 92                                                      0026500
 5020515000 0 91 92                                                      0026600
 5020516000 0 91 92                                                      0026700
 5020517000 0 91 92                                                      0026800
 5020518000 0 91 92                                                      0026900
 5020519000 0 91 92                                                      0027000
 5020520000 0 91 92                                                      0027100
 5020521000 0 91 92                                                      0027200
 5020522000 0 91 92                                                      0027300
 5020523000 0 91 92                                                      0027400
 5020524000 0 91 92                                                      0027500
 5020525000 0 91 92                                                      0027600
 5020526000 0 91 92                                                      0027700
 5020527000 0 91 92                                                      0027800
 5020528000 0 91 92                                                      0027900
 5020529000 0 91 92                                                      0028000
 5020530000 0 91 92                                                      0028100
 5020531000 0 91 92                                                      0028200
 5020532000 0 91 92                                                      0028300
 5020533000 0 91 92                                                      0028400
 5020534000 0 91 92                                                      0028500
 5020535000 0 91 92                                                      0028600
 5020536000 0 91 92                                                      0028700
 5020537000 0 91 92                                                      0028800
 5020538000 0 91 92                                                      0028900
 5020539000 0 91 92                                                      0029000
 5020540000 0 91 92                                                      0029100
 5020541000 0 91 92                                                      0029200
 5020542000 0 91 92                                                      0029300
 5020543000 0 91 92                                                      0029400
 5020544000 0 91 92                                                      0029500
 5020545000 0 91 92                                                      0029600
 5020546000 0 91 92                                                      0029700
 5020547000 0 91 92                                                      0029800
 5020548000 0 91 92                                                      0029900
 5020549000 0 91 92                                                      0030000
 5020550000 0 91 92                                                      0030100
 5020551000 0 91 92                                                      0030200
 5020552000 0 91 92                                                      0030300
 5020553000 0 91 92                                                      0030400
 5020554000 0 91 92                                                      0030500
 5020555000 0 91 92                                                      0030600
 5020556000 0 91 92                                                      0030700
 5020557000 0 91 92                                                      0030800
 5020558000 0 91 92                                                      0030900
 5020559000 0 91 92                                                      0031000
 5020560000 0 91 92                                                      0031100
 5020561000 0 91 92                                                      0031200
 5020562000 0 91 92                                                      0031300
 5020563000 0 91 92                                                      0031400
 5020564000 0 91 92                                                      0031500
 5020565000 0 91 92                                                      0031600
 5020566000 0 91 92                                                      0031700
 5020567000 0 91 92                                                      0031800
 5020568000 0 91 92                                                      0031900
 5020569000 0 91 92                                                      0032000
 5020570000 0 91 92                                                      0032100
 5020571000 0 91 92                                                      0032200
 5020572000 0 91 92                                                      0032300
 5020573000 0 91 92                                                      0032400
 5020574000 0 91 92                                                      0032500
 5020575000 0 91 92                                                      0032600
 5020576000 0 91 92                                                      0032700
 5020577000 0 91 92                                                      0032800
 5020578000 0 91 92                                                      0032900
 5020579000 0 91 92                                                      0033000
 5020580000 0 91 92                                                      0033100
 5020581000 0 91 92                                                      0033200
 5020582000 0 91 92                                                      0033300
 5020583000 0 91 92                                                      0033400
 5020584000 0 91 92                                                      0033500
 5020585000 0 91 92                                                      0033600
 5020586000 0 91 92                                                      0033700
 5020587000 0 91 92                                                      0033800
 5020588000 0 91 92                                                      0033900
 5020589000 0 91 92                                                      0034000
 5020590000 0 91 92                                                      0034100
 5020591000 0 91 92                                                      0034200
 5020592000 0 91 92                                                      0034300
 5020593000 0 91 92                                                      0034400
 5020594000 0 91 92                                                      0034500
 5020595000 0 91 92                                                      0034600
 5020596000 0 91 92                                                      0034700
 5020597000 0 91 92                                                      0034800
 5020598000 0 91 92                                                      0034900
 5020599000 0 91 92                                                      0035000
 5020600000 0 91 92                                                      0035100
 5020601000 0 91 92                                                      0035200
 5020602000 0 91 92                                                      0035300
 5020603000 0 91 92                                                      0035400
 5020604000 0 91 92                                                      0035500
 5020605000 0 91 92                                                      0035600
 5020606000 0 91 92                                                      0035700
 5020607000 0 91 92                                                      0035800
 5020608000 0 91 92                                                      0035900
 5020609000 0 91 92                                                      0036000
 5020610000 0 91 92                                                      0036100
 5020611000 0 91 92                                                      0036200
 5020612000 0 91 92                                                      0036300
 5020613000 0 91 92                                                      0036400
 5020614000 0 91 92                                                      0036500
 5020615000 0 91 92                                                      0036600
 5020616000 0 91 92                                                      0036700
 5020617000 0 91 92                                                      0036800
 5020618000 0 91 92                                                      0036900
 5020619000 0 91 92                                                      0037000
 5020620000 0 91 92                                                      0037100
 5020621000 0 91 92                                                      0037200
 5020622000 0 91 92                                                      0037300
 5020623000 0 91 92                                                      0037400
 5020624000 0 91 92                                                      0037500
 5020625000 0 91 92                                                      0037600
 5020626000 0 91 92                                                      0037700
 5020627000 0 91 92                                                      0037800
 5020628000 0 91 92                                                      0037900
 5020629000 0 91 92                                                      0038000
 5020630000 0 91 92                                                      0038100
 5020631000 0 91 92                                                      0038200
 5020632000 0 91 92                                                      0038300
 5020633000 0 91 92                                                      0038400
 5020634000 0 91 92                                                      0038500
 5020635000 0 91 92                                                      0038600
 5020636000 0 91 92                                                      0038700
 5020637000 0 91 92                                                      0038800
 5020638000 0 91 92                                                      0038900
 5020639000 0 91 92                                                      0039000
 5020640000 0 91 92                                                      0039100
 5020641000 0 91 92                                                      0039200
 5020642000 0 91 92                                                      0039300
 5020643000 0 91 92                                                      0039400
 5020644000 0 91 92                                                      0039500
 5020645000 0 91 92                                                      0039600
 5020646000 0 91 92                                                      0039700
 5020647000 0 91 92                                                      0039800
 5020648000 0 91 92                                                      0039900
 5020649000 0 91 92                                                      0040000
 5020650000 0 91 92                                                      0040100
 5020651000 0 91 92                                                      0040200
 5020652000 0 91 92                                                      0040300
 5020653000 0 91 92                                                      0040400
 5020654000 0 91 92                                                      0040500
 5020655000 0 91 92                                                      0040600
 5020656000 0 91 92                                                      0040700
 5020657000 0 91 92                                                      0040800
 5020658000 0 91 92                                                      0040900
 5020659000 0 91 92                                                      0041000
 5020660000 0 91 92                                                      0041100
 5020661000 0 91 92                                                      0041200
 5020662000 0 91 92                                                      0041300
 5020663000 0 91 92                                                      0041400
 5020664000 0 91 92                                                      0041500
 5020665000 0 91 92                                                      0041600
 5020666000 0 91 92                                                      0041700
 5020667000 0 91 92                                                      0041800
 5020668000 0 91 92                                                      0041900
 5020669000 0 91 92                                                      0042000
 5020670000 0 91 92                                                      0042100
 5020671000 0 91 92                                                      0042200
 5020672000 0 91 92                                                      0042300
 5020673000 0 91 92                                                      0042400
 5020674000 0 91 92                                                      0042500
 5020675000 0 91 92                                                      0042600
 5020676000 0 91 92                                                      0042700
 5020677000 0 91 92                                                      0042800
 5020678000 0 91 92                                                      0042900
 5020679000 0 91 92                                                      0043000
 5020680000 0 91 92                                                      0043100
 5020681000 0 91 92                                                      0043200
 5020682000 0 91 92                                                      0043300
 5020683000 0 91 92                                                      0043400
 5020684000 0 91 92                                                      0043500
 5020685000 0 91 92                                                      0043600
 5020686000 0 91 92                                                      0043700
 5020687000 0 91 92                                                      0043800
 5020688000 0 91 92                                                      0043900
 5020689000 0 91 92                                                      0044000
 5020690000 0 91 92                                                      0044100
 5020691000 0 91 92                                                      0044200
 5020692000 0 91 92                                                      0044300
 5020693000 0 91 92                                                      0044400
 5020694000 0 91 92                                                      0044500
 5020695000 0 91 92                                                      0044600
 5020696000 0 91 92                                                      0044700
 5020697000 0 91 92                                                      0044800
 5020698000 0 91 92                                                      0044900
 5020699000 0 91 92                                                      0045000
 5020700000 0 91 92                                                      0045100
 5020701000 0 91 92                                                      0045200
 5020702000 0 91 92                                                      0045300
 5020703000 0 91 92                                                      0045400
 5020704000 0 91 92                                                      0045500
 5020705000 0 91 92                                                      0045600
 5020706000 0 91 92                                                      0045700
 5020707000 0 91 92                                                      0045800
 5020708000 0 91 92                                                      0045900
 5020709000 0 91 92                                                      0046000
 5020710000 0 91 92                                                      0046100
 5020711000 0 91 92                                                      0046200
 5020712000 0 91 92                                                      0046300
 5020713000 0 91 92                                                      0046400
 5020714000 0 91 92                                                      0046500
 5020715000 0 91 92                                                      0046600
 5020716000 0 91 92                                                      0046700
 5020717000 0 91 92                                                      0046800
 5020718000 0 91 92                                                      0046900
 5020719000 0 91 92                                                      0047000
 5020720000 0 91 92                                                      0047100
 5020721000 0 91 92                                                      0047200
 5020722000 0 91 92                                                      0047300
 5020723000 0 91 92                                                      0047400
 5020724000 0 91 92                                                      0047500
 5020725000 0 91 92                                                      0047600
 5020726000 0 91 92                                                      0047700
 5020727000 0 91 92                                                      0047800
 5020728000 0 91 92                                                      0047900
 5020729000 0 91 92                                                      0048000
 5020730000 0 91 92                                                      0048100
 5020731000 0 91 92                                                      0048200
 5020732000 0 91 92                                                      0048300
 5020733000 0 91 92                                                      0048400
 5020734000 0 91 92                                                      0048500
 5020735000 0 91 92                                                      0048600
 5020736000 0 91 92                                                      0048700
 5020737000 0 91 92                                                      0048800
 5020738000 0 91 92                                                      0048900
 5020739000 0 91 92                                                      0049000
 5020740000 0 91 92                                                      0049100
 5020741000 0 91 92                                                      0049200
 5020742000 0 91 92                                                      0049300
 5020743000 0 91 92                                                      0049400
 5020744000 0 91 92                                                      0049500
 5020745000 0 91 92                                                      0049600
 5020746000 0 91 92                                                      0049700
 5020747000 0 91 92                                                      0049800
 5020748000 0 91 92                                                      0049900
 5020749000 0 91 92                                                      0050000
 5020750000 0 91 92                                                      0050100
 5020751000 0 91 92                                                      0050200
 5020752000 0 91 92                                                      0050300
 5020753000 0 91 92                                                      0050400
 5020754000 0 91 92                                                      0050500
 5020755000 0 91 92                                                      0050600
 5020756000 0 91 92                                                      0050700
 5020757000 0 91 92                                                      0050800
 5020758000 0 91 92                                                      0050900
 5020759000 0 91 92                                                      0051000
 5020760000 0 91 92                                                      0051100
 5020761000 0 91 92                                                      0051200
 5020762000 0 91 92                                                      0051300
 5020763000 0 91 92                                                      0051400
 5020764000 0 91 92                                                      0051500
 5020765000 0 91 92                                                      0051600
 5020766000 0 91 92                                                      0051700
 5020767000 0 91 92                                                      0051800
 5020768000 0 91 92                                                      0051900
 5020769000 0 91 92                                                      0052000
 5020770000 0 91 92                                                      0052100
 5020771000 0 91 92                                                      0052200
 5020772000 0 91 92                                                      0052300
 5020773000 0 91 92                                                      0052400
 5020774000 0 91 92                                                      0052500
 5020775000 0 91 92                                                      0052600
 5020776000 0 91 92                                                      0052700
 5020777000 0 91 92                                                      0052800
 5020778000 0 91 92                                                      0052900
 5020779000 0 91 92                                                      0053000
 5020780000 0 91 92                                                      0053100
 5020781000 0 91 92                                                      0053200
 5020782000 0 91 92                                                      0053300
 5020783000 0 91 92                                                      0053400
 5020784000 0 91 92                                                      0053500
 5020785000 0 91 92                                                      0053600
 5020786000 0 91 92                                                      0053700
 5020787000 0 91 92                                                      0053800
 5020788000 0 91 92                                                      0053900
 5020789000 0 91 92                                                      0054000
 5020790000 0 91 92                                                      0054100
 5020791000 0 91 92                                                      0054200
 5020792000 0 91 92                                                      0054300
 5020793000 0 91 92                                                      0054400
 5020794000 0 91 92                                                      0054500
 5020795000 0 91 92                                                      0054600
 5020796000 0 91 92                                                      0054700
 5020797000 0 91 92                                                      0054800
 5020798000 0 91 92                                                      0054900
 5020799000 0 91 92                                                      0055000
 5020800000 0 91 92                                                      0055100
 5020801000 0 91 92                                                      0055200
 5020802000 0 91 92                                                      0055300
 5020803000 0 91 92                                                      0055400
 5020804000 0 91 92                                                      0055500
 5020805000 0 91 92                                                      0055600
 5020806000 0 91 92                                                      0055700
 5020807000 0 91 92                                                      0055800
 5020808000 0 91 92                                                      0055900
 5020809000 0 91 92                                                      0056000
 5020810000 0 91 92                                                      0056100
 5020811000 0 91 92                                                      0056200
 5020812000 0 91 92                                                      0056300
 5020813000 0 91 92                                                      0056400
 5020814000 0 91 92                                                      0056500
 5020815000 0 91 92                                                      0056600
 5020816000 0 91 92                                                      0056700
 5020817000 0 91 92                                                      0056800
 5020818000 0 91 92                                                      0056900
 5020819000 0 91 92                                                      0057000
 5020820000 0 91 92                                                      0057100
 5020821000 0 91 92                                                      0057200
 5020822000 0 91 92                                                      0057300
 5020823000 0 91 92                                                      0057400
 5020824000 0 91 92                                                      0057500
 5020825000 0 91 92                                                      0057600
 5020826000 0 91 92                                                      0057700
 5020827000 0 91 92                                                      0057800
 5020828000 0 91 92                                                      0057900
 5020829000 0 91 92                                                      0058000
 5020830000 0 91 92                                                      0058100
 5020831000 0 91 92                                                      0058200
 5020832000 0 91 92                                                      0058300
 5020833000 0 91 92                                                      0058400
 5020834000 0 91 92                                                      0058500
 5020835000 0 91 92                                                      0058600
 5020836000 0 91 92                                                      0058700
 5020837000 0 91 92                                                      0058800
 5020838000 0 91 92                                                      0058900
 5020839000 0 91 92                                                      0059000
 5020840000 0 91 92                                                      0059100
 5020841000 0 91 92                                                      0059200
 5020842000 0 91 92                                                      0059300
 5020843000 0 91 92                                                      0059400
 5020844000 0 91 92                                                      0059500
 5020845000 0 91 92                                                      0059600
 5020846000 0 91 92                                                      0059700
 5020847000 0 91 92                                                      0059800
 5020848000 0 91 92                                                      0059900
 5020849000 0 91 92                                                      0060000
 5020850000 0 91 92                                                      0060100
 5020851000 0 91 92                                                      0060200
 5020852000 0 91 92                                                      0060300
 5020853000 0 91 92                                                      0060400
 5020854000 0 91 92                                                      0060500
 5020855000 0 91 92                                                      0060600
 5020856000 0 91 92                                                      0060700
 5020857000 0 91 92                                                      0060800
 5020858000 0 91 92                                                      0060900
 5020859000 0 91 92                                                      0061000
 5020860000 0 91 92                                                      0061100
 5020861000 0 91 92                                                      0061200
 5020862000 0 91 92                                                      0061300
 5020863000 0 91 92                                                      0061400
 5020864000 0 91 92                                                      0061500
 5020865000 0 91 92                                                      0061600
 5020866000 0 91 92                                                      0061700
 5020867000 0 91 92                                                      0061800
 5020868000 0 91 92                                                      0061900
 5020869000 0 91 92                                                      0062000
 5020870000 0 91 92                                                      0062100
 5020871000 0 91 92                                                      0062200
 5020872000 0 91 92                                                      0062300
 5020873000 0 91 92                                                      0062400
 5020874000 0 91 92                                                      0062500
 5020875000 0 91 92                                                      0062600
 5020876000 0 91 92                                                      0062700
 5020877000 0 91 92                                                      0062800
 5020878000 0 91 92                                                      0062900
 5020879000 0 91 92                                                      0063000
 5020880000 0 91 92                                                      0063100
 5020881000 0 91 92                                                      0063200
 5020882000 0 91 92                                                      0063300
 5020883000 0 91 92                                                      0063400
 5020884000 0 91 92                                                      0063500
 5020885000 0 91 92                                                      0063600
 5020886000 0 91 92                                                      0063700
 5020887000 0 91 92                                                      0063800
 5020888000 0 91 92                                                      0063900
 5020889000 0 91 92                                                      0064000
 5020890000 0 91 92                                                      0064100
 5020891000 0 91 92                                                      0064200
 5020892000 0 91 92                                                      0064300
 5020893000 0 91 92                                                      0064400
 5020894000 0 91 92                                                      0064500
 5020895000 0 91 92                                                      0064600
 5020896000 0 91 92                                                      0064700
 5020897000 0 91 92                                                      0064800
 5020898000 0 91 92                                                      0064900
 5020899000 0 91 92                                                      0065000
 5020900000 0 91 92                                                      0065100
 5020901000 0 91 92                                                      0065200
 5020902000 0 91 92                                                      0065300
 5020903000 0 91 92                                                      0065400
 5020904000 0 91 92                                                      0065500
 5020905000 0 91 92                                                      0065600
 5020906000 0 91 92                                                      0065700
 5020907000 0 91 92                                                      0065800
 5020908000 0 91 92                                                      0065900
 5020909000 0 91 92                                                      0066000
 5020910000 0 91 92                                                      0066100
 5020911000 0 91 92                                                      0066200
 5020912000 0 91 92                                                      0066300
 5020913000 0 91 92                                                      0066400
 5020914000 0 91 92                                                      0066500
 5020915000 0 91 92                                                      0066600
 5020916000 0 91 92                                                      0066700
 5020917000 0 91 92                                                      0066800
 5020918000 0 91 92                                                      0066900
 5020919000 0 91 92                                                      0067000
 5020920000 0 91 92                                                      0067100
 5020921000 0 91 92                                                      0067200
 5020922000 0 91 92                                                      0067300
 5020923000 0 91 92                                                      0067400
 5020924000 0 91 92                                                      0067500
 5020925000 0 91 92                                                      0067600
 5020926000 0 91 92                                                      0067700
 5020927000 0 91 92                                                      0067800
 5020928000 0 91 92                                                      0067900
 5020929000 0 91 92                                                      0068000
 5020930000 0 91 92                                                      0068100
 5020931000 0 91 92                                                      0068200
 5020932000 0 91 92                                                      0068300
 5020933000 0 91 92                                                      0068400
 5020934000 0 91 92                                                      0068500
 5020935000 0 91 92                                                      0068600
 5020936000 0 91 92                                                      0068700
 5020937000 0 91 92                                                      0068800
 5020938000 0 91 92                                                      0068900
 5020939000 0 91 92                                                      0069000
 5020940000 0 91 92                                                      0069100
 5020941000 0 91 92                                                      0069200
 5020942000 0 91 92                                                      0069300
 5020943000 0 91 92                                                      0069400
 5020944000 0 91 92                                                      0069500
 5020945000 0 91 92                                                      0069600
 5020946000 0 91 92                                                      0069700
 5020947000 0 91 92                                                      0069800
 5020948000 0 91 92                                                      0069900
 5020949000 0 91 92                                                      0070000
 5020950000 0 91 92                                                      0070100
 5020951000 0 91 92                                                      0070200
 5020952000 0 91 92                                                      0070300
 5020953000 0 91 92                                                      0070400
 5020954000 0 91 92                                                      0070500
 5020955000 0 91 92                                                      0070600
 5020956000 0 91 92                                                      0070700
 5020957000 0 91 92                                                      0070800
 5020958000 0 91 92                                                      0070900
 5020959000 0 91 92                                                      0071000
 5020960000 0 91 92                                                      0071100
 5020961000 0 91 92                                                      0071200
 5020962000 0 91 92                                                      0071300
 5020963000 0 91 92                                                      0071400
 5020964000 0 91 92                                                      0071500
 5020965000 0 91 92                                                      0071600
 5020966000 0 91 92                                                      0071700
 5020967000 0 91 92                                                      0071800
 5020968000 0 91 92                                                      0071900
 5020969000 0 91 92                                                      0072000
 5020970000 0 91 92                                                      0072100
 5020971000 0 91 92                                                      0072200
 5020972000 0 91 92                                                      0072300
 5020973000 0 91 92                                                      0072400
 5020974000 0 91 92                                                      0072500
 5020975000 0 91 92                                                      0072600
 5020976000 0 91 92                                                      0072700
 5020977000 0 91 92                                                      0072800
 5020978000 0 91 92                                                      0072900
 5020979000 0 91 92                                                      0073000
 5020980000 0 91 92                                                      0073100
 5020981000 0 91 92                                                      0073200
 5020982000 0 91 92                                                      0073300
 5020983000 0 91 92                                                      0073400
 5020984000 0 91 92                                                      0073500
 5020985000 0 91 92                                                      0073600
 5020986000 0 91 92                                                      0073700
 5020987000 0 91 92                                                      0073800
 5020988000 0 91 92                                                      0073900
 5020989000 0 91 92                                                      0074000
 5020990000 0 91 92                                                      0074100
 5020991000 0 91 92                                                      0074200
 5020992000 0 91 92                                                      0074300
 5020993000 0 91 92                                                      0074400
 5020994000 0 91 92                                                      0074500
 5020995000 0 91 92                                                      0074600
 5020996000 0 91 92                                                      0074700
 5020997000 0 91 92                                                      0074800
 5020998000 0 91 92                                                      0074900
 5020999000 0 91 92                                                      0075000
```

```
 45  12   2  1  26                                        0000   001000
102   3   4  1 201                                    00000360   001010
103   3   4  1 201                                    00000370   001020
502040300       0     0 47   128   9                  00000380   001030
 61   1   2  0                                                   001040
 90   2   2  2  91   92                                          001050
 99   1   2  0                                                   001060
 15  12   2  1  26                                        0000   001070
 25  12   2  1  26                                        0000   001080
 35  12   2  1  26                                        0000   001090
 45  12   2  1  26                                        0000   001100
102   3   4  1 201                                    00000430   001110
103   3   4  1 201                                    00000440   001120
502040400       0     0 47   128   9                  00000450   001130
 61   1   2  0                                                   001140
 90   2   2  2  91   92                                          001150
 99   1   2  0                                                   001160
 15  12   2  1  26                                        0000   001170
 25  12   2  1  26                                        0000   001180
 35  12   2  1  26                                        0000   001190
 45  12   2  1  26                                        0000   001200
102   3   4  1 201                                    00000500   001210
103   3   4  1 201                                    00000510   001220
502040500       0     0 47   128   9                  00000520   001230
 61   1   2  0                                                   001240
 90   2   2  2  91   92                                          001250
 99   1   2  0                                                   001260
 15  12   2  1  26                                        0000   001270
 25  12   2  1  26                                        0000   001280
 35  12   2  1  26                                        0000   001290
 45  12   2  1  26                                        0000   001300
102   3   4  1 201                                    00000570   001310
103   3   4  1 201                                    00000580   001320
502040600       0     0 47   128   9                  00000590   001330
 61   1   2  0                                                   001340
 90   2   2  2  91   92                                          001350
 99   1   2  0                                                   001360
 15  12   2  1  26                                        0000   001370
 25  12   2  1  26                                        0000   001380
 35  12   2  1  26                                        0000   001390
 45  12   2  1  26                                        0000   001400
102   3   4  1 201                                    00000640   001410
103   3   4  1 201                                    00000650   001420
502040700       0     0 47   128   9                  00000660   001430
 61   1   2  0                                                   001440
 90   2   2  2  91   92                                          001450
 99   1   2  0                                                   001460
 15  12   2  1  26                                        0000   001470
 25  12   2  1  26                                        0000   001480
 35  12   2  1  26                                        0000   001490
 45  12   2  1  26                                        0000   001500
102   3   4  1 201                                    00000710   001510
103   3   4  1 201                                    00000720   001520
502040800       0     0 47   128   9                  00000730   001530
 61   1   2  0                                                   001540
 90   2   2  2  91   92                                          001550
```

LISTING OF MODULE DODATA

```
  103    3  4 1 201                                                              00001140   002120
++END OF COMMAND                                                                 00001150   002130
++COMMAND=DEFINE         ++OPERAND=DIRECTORY                                     0000000    002140
    1   8    550 -550 USSR                                  -1   -1    2         800000000  002150
    2   1    501 -359 WESTERN                                1   12    3         801000000  002160
    3   1    554 -250 BALTICS                                2    4   -1         801010000  002170
    3   2    557 -257 BELORUSSIA                             2    5   -1         801020000  002180
    3   3    500 -252 WEST UKRAINE                           2    6   -1         801030000  002190
    3   4    499 -302 NORTH CENTRAL UKRAINE                  2    7   -1         801040000  002200
    3   5    500 -350 NORTHEAST UKRAINE                      2    8   -1         801050000  002210
    3   6    455 -353 EASTERN UKRAINE                        2    9   -1         801060000  002220
    3   7    453 -303 SOUTHERN UKRAINE                       2   10   -1         801070000  002230
    3   8    450 -258 MOLDAVIA                               2   11   -1         801080000  002240
    3   9    450 -400 KRASNODAR                              2   -1   -1         801090000  002250
    2   2    506 -450 CENTRAL                                1   22   13         802000000  002260
    3  10    452 -408 NORTHEAST CAUCACUS                    12   14   -1         802100000  002270
    3  11    504 -351 WEST BLACK SOIL ZONE                  12   15   -1         802110000  002280
    3  12    504 -402 EAST BLACK SOIL ZONE                  12   16   -1         802120000  002290
    3  13    551 -355 CENTRAL REGION                        12   17   -1         802130000  002300
    3  14    552 -454 VOLGA-VYATSK                          12   18   -1         802140000  002310
    3  15    509 -500 UPPER VOLGA                           12   19   -1         802150000  002320
    3  16    502 -455 MIDDLE VOLGA                          12   20   -1         802160000  002330
    3  17    458 -450 LOWER VOLGA                           12   21   -1         802170000  002340
    3  18    555 -551 NORTHWEST URALS                       12   -1   -1         802180000  002350
    2   3    551 -701 EAST CENTRAL                           1   -1   23         803000000  002360
    3  19    506 -552 SOUTHERN URALS                        22   24   -1         803190000  002370
    3  20    502 -700 NORTHEASTERN URALS                    22   25   -1         803200000  002380
    3  21    455 -550 WESTERN KAZAKHSTAN                    22   26   -1         803210000  002390
    3  22    552 -608 KUSTANAY                              22   27   -1         803220000  002400
    3  23    502 -658 TSELINOGRAD                           22   28   -1         803230000  002410
    3  24    506 -659 NORTHERN KAZAKHSTAN                   22   29   -1         803240000  002420
    3  25    502 -752 PAVLODAR                              22   30   -1         803250000  002430
    3  26    552 -800 WESTERN SIBERIA                       22   31   -1         803260000  002440
    3  27    504 -807 ALTAI KRAY                            22   -1   -1         803270000  002450
++END OF COMMAND                                                                 0000000    002460
++COMMAND=DEFINE         ++OPERAND=DESCRIPTOR                                    00000020   002470
   801010000         0     0 22    88  7                                         00000030   002480
   61    1   2 6                                                                 00000040   002490
   90    2   2 2    91   92                                                      00000050   002500
   99    1   2 6                                                                 00000060   002510
   35   12   2 1    20                                                           00000070   002520
    5   12   2 1    20                                                           00000080   002530
  101    4   4 1   202                                                           00000090   002540
  103    4   4 1   202                                                           00000100   002550
   801020000         0     0 22    88  7                                         00000110   002560
   61    1   2 6                                                                 00000120   002570
   90    2   2 2    91   92                                                      00000130   002580
   99    1   2 6                                                                 00000140   002590
   35   12   2 1    20                                                           00000150   002600
    5   12   2 1    20                                                           00000160   002610
  101    4   4 1   202                                                           00000170   002620
  103    4   4 1   202                                                           00000180   002630
   801030000         0     0 22    88  7                                         00000190   002640
   61    1   2 6                                                                 00000200   002650
   90    2   2 2    9    92                                                      00000210   002660
   99    1   2 6                                                                 00000220   002670
```

117

RUN NO. 7 DATE 10/28/75 TIME 1106 LISTING OF MODULE OBDATA

```
 35   12  2  1  26                                   00000230  002680
  5   12  2  1  26                                   00000240  002690
101    4  4  1  202                                  00000250  002700
103    4  4  1  202                                  00000260  002710
 801040000       0    0 22   88   7                  00000270  002720
 61    1  2  0                                       00000280  002730
 90    2  2  2  91   92                              00000290  002740
 99    1  2  0                                       00000300  002750
 35   12  2  1  26                                   00000310  002760
  5   12  2  1  26                                   00000320  002770
101    4  4  1  202                                  00000330  002780
103    4  4  1  202                                  00000340  002790
 801050000       0    0 22   88   7                  00000350  002800
 61    1  2  0                                       00000360  002810
 90    2  2  2  91   92                              00000370  002820
 99    1  2  0                                       00000380  002830
 35   12  2  1  26                                   00000390  002840
  5   12  2  1  26                                   00000400  002850
101    4  4  1  202                                  00000410  002860
103    4  4  1  202                                  00000420  002870
 801060000       0    0 22   88   7                  00000430  002880
 61    1  2  0                                       00000440  002890
 90    2  2  2  91   92                              00000450  002900
 99    1  2  0                                       00000460  002910
 35   12  2  1  26                                   00000470  002920
  5   12  2  1  26                                   00000480  002930
101    4  4  1  202                                  00000490  002940
103    4  4  1  202                                  00000500  002950
 801070000       0    0 22   88   7                  00000510  002960
 61    1  2  0                                       00000520  002970
 90    2  2  2  91   92                              00000530  002980
 99    1  2  0                                       00000540  002990
 35   12  2  1  26                                   00000550  003000
  5   12  2  1  26                                   00000560  003010
101    4  4  1  202                                  00000570  003020
103    4  4  1  202                                  00000580  003030
 801080000       0    0 22   88   7                  00000590  003040
 61    1  2  0                                       00000600  003050
 90    2  2  2  91   92                              00000610  003060
 99    1  2  0                                       00000620  003070
 35   12  2  1  26                                   00000630  003080
  5   12  2  1  26                                   00000640  003090
101    4  4  1  202                                  00000650  003100
103    4  4  1  202                                  00000660  003110
 801090000       0    0 22   88   7                  00000670  003120
 61    1  2  0                                       00000680  003130
 90    2  2  2  91   92                              00000690  003140
 99    1  2  0                                       00000700  003150
 35   12  2  1  26                                   00000710  003160
  5   12  2  1  26                                   00000720  003170
101    4  4  1  202                                  00000730  003180
103    4  4  1  202                                  00000740  003190
 801100000       0    0 22   88   7                  00000750  003200
 61    1  2  0                                       00000760  003210
 90    2  2  2  91   92                              00000770  003220
 99    1  2  0                                       00000780  003230
```

RUN NO. 7 DATE 10/28/76 TIME 1106 LISTING OF MODULE DBDATA

```
 35   12   2  1   26                                  00000790  003240
  5   12   2  1   26                                  00000800  003250
101    4   4  2  201  202                             00000810  003260
103    4   4  2  201  202                             00000820  003270
  802110000        0      0 22    88   7              00000830  003280
 61    1   2  0                                       00000840  003290
 98    2   2  2   91   92                             00000850  003300
 99    1   2  0                                       00000860  003310
 35   12   2  1   26                                  00000870  003320
  5   12   2  1   26                                  00000880  003330
101    4   4  2  201  202                             00000890  003340
103    4   4  2  201  202                             00000900  003350
  802120000        0      0 22    88   7              00000910  003360
 61    1   2  0                                       00000920  003370
 98    2   2  2   91   92                             00000930  003380
 99    1   2  0                                       00000940  003390
 35   12   2  1   26                                  00000950  003400
  5   12   2  1   26                                  00000960  003410
101    4   4  2  201  202                             00000970  003420
103    4   4  2  201  202                             00000980  003430
  802130000        0      0 22    88   7              00000990  003440
 61    1   2  0                                       00001000  003450
 98    2   2  2   91   92                             00001010  003460
 99    1   2  0                                       00001020  003470
 35   12   2  1   26                                  00001030  003480
  5   12   2  1   26                                  00001040  003490
101    4   4  2  201  202                             00001050  003500
103    4   4  2  201  202                             00001060  003510
  802140000        0      0 22    88   7              00001070  003520
 61    1   2  0                                       00001080  003530
 98    2   2  2   91   92                             00001090  003540
 99    1   2  0                                       00001100  003550
 35   12   2  1   26                                  00001110  003560
  5   12   2  1   26                                  00001120  003570
101    4   4  2  201  202                             00001130  003580
103    4   4  2  201  202                             00001140  003590
  802150000        0      0 22    88   7              00001150  003600
 61    1   2  0                                       00001160  003610
 98    2   2  2   91   92                             00001170  003620
 99    1   2  0                                       00001180  003630
 35   12   2  1   26                                  00001190  003640
  5   12   2  1   26                                  00001200  003650
101    4   4  2  201  202                             00001210  003660
103    4   4  2  201  202                             00001220  003670
  802160000        0      0 22    88   7              00001230  003680
 61    1   2  0                                       00001240  003690
 98    2   2  2   91   92                             00001250  003700
 99    1   2  0                                       00001260  003710
 35   12   2  1   26                                  00001270  003720
  5   12   2  1   26                                  00001280  003730
101    4   4  2  201  202                             00001290  003740
103    4   4  2  201  202                             00001300  003750
  802170000        0      0 22    88   7              00001310  003760
 61    1   2  0                                       00001320  003770
 98    2   2  2   91   92                             00001330  003780
 99    1   2  0                                       00001340  003790
```

RUN NO. 7 DATE 10/28/71 TIME 1100

LISTING OF MODULE OBDATA

```
00001350  003800
00001360  003810
00001370  003820
00001380  003830
00001390  003840
00001400  003850
00001410  003860
00001420  003870
00001430  003880
00001440  003890
00001450  003900
00001460  003910
00001470  003920
00001480  003930
00001490  003940
00001500  003950
00001510  003960
00001520  003970
00001530  003980
00001540  003990
00001550  004000
00001560  004010
00001570  004020
00001580  004030
00001590  004040
00001600  004050
00001610  004060
00001620  004070
00001630  004080
00001640  004090
00001650  004100
00001660  004110
00001670  004120
00001680  004130
00001690  004140
00001700  004150
00001710  004160
00001720  004170
00001730  004180
00001740  004190
00001750  004200
00001760  004210
00001770  004220
00001780  004230
00001790  004240
00001800  004250
00001810  004260
00001820  004270
00001830  004280
00001840  004290
00001850  004300
00001860  004310
00001870  004320
00001880  004330
00001890  004340
00001900  004350
```

LISTING OF SOURCE DATA

```
0000360
0004370
0004380
0004390
0004400
0004410
0004420
0004430
0004440
0004450
0004460
0004470
0004480
0004490
0004500
0004510
0004520
0004530
0004540
0004550
0004560
0004570
0004580
0004590
0004600
0004610
0004620
0004630
0004640
0004650
```